ARTICLE

Processing でプログラミングに挑戦！ —第 1 回 図形を描いてみよう—

杉浦　学
鎌倉女子大学

この連載について

この連載では，プログラミングの経験が少ないジュニア会員を対象に，プログラミングの入門記事を 4 回にわたって掲載していきます．命令をマウスで組み立てる（Scratch のような）タイプのプログラミング経験があり，命令を文字で入力することに挑戦したいというジュニア会員の皆様（主に中学生〜高校生）を読者として想定しました．

記事の中には「練習問題」や次号までの「宿題」を用意してあります．文章を眺めるだけでなく，実際に手と頭を動かしながら読んでみてください．

Processing とは

プログラミングを簡単に説明すれば「コンピュータが理解できる言葉を使って，実行してほしいことをコンピュータに伝えること」といえるでしょう．

この連載では，コンピュータが理解できる言葉として，「Processing（プロセッシング）」を利用します．Processing を使えば，画面に図形を表示して動かしたり，ユーザの操作に反応したりする仕組みを手軽に作ることができます．

Processing は Java というプログラミング言語をもとにして作られています．短い命令でいろいろなことができるように工夫されており，プログラミングの初心者でも扱いが簡単です．Processing の生い立ちなどを詳しく知りたい場合は，文献 1）の『Processing をはじめよう 第 2 版』を読んでみてください．

準備

さっそく Processing を使ったプログラミングの準備をはじめましょう．Processing の公式サイトのダウンロードページ（https://processing.org/download/）にアクセスし，お使いの PC のオペレーティングシステム（OS）にあったファイルをダウンロードしてください（図 -1）．Windows で使う場合の手順を簡単に解説していきます．

図 -1　OS にあったファイルをダウンロード

Windows の場合は「processing-X.X.X-windows64.zip」という圧縮ファイル（X.X.X はバージョン番号）がダウンロードされるので，これを展開します．展開が完了すると「processing-X.X.X」というフォルダができるので，分かりやすいところ（たとえばデスクトップなど）に移動しておきましょう．フォルダの中の「processing.exe」をダブルクリックすれば，Processing が起動します（図 -2）．設定によってはセキュリティの警告が表示される場合がありますが，その場合は「アクセスを許可する」を選択してください．

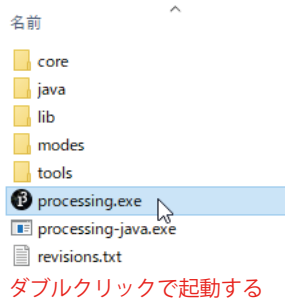


図 -2　Processing の起動

初回の起動時には「Welcome to Processing 3」と書かれた小さなウィンドウが表示されます．「Show this message on start up」のチェックを外してから，「Get Started」をクリックします（図 -3）．

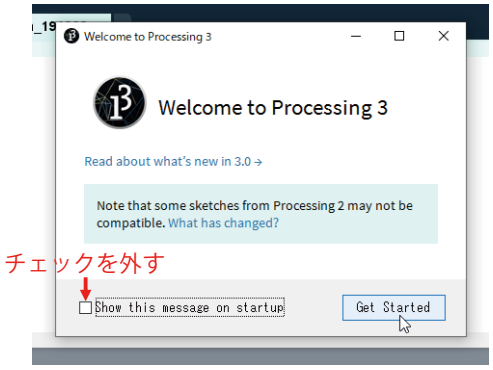


図 -3　初回起動時の設定

Processing を起動した直後の画面を図 -4 に示します．

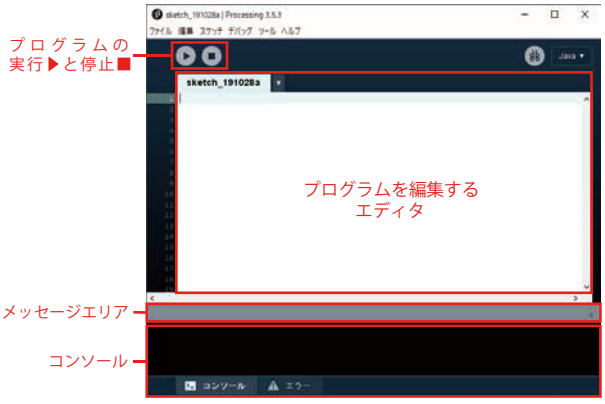


図 -4　起動直後の画面と各部分の名称

最初に日本語の入力と表示ができるように設定しておきましょう．ファイルメニューから「設定」をクリックし，「エディタとコンソールのフォント」を日本語が表示できるものに変更しておきます（図 -5）．エディタのフォントサイズも少し大きめ（たとえば 14）にしておくとよいでしょう．画面の例では「MS ゴシック」の「14」ポイントに設定しています．

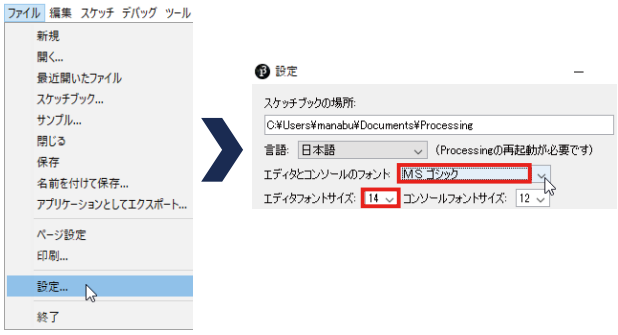


図 -5　フォントとサイズの設定

さっそくプログラムを書いて，実行してみましょう．Processing ではプログラムのことを「スケッチ（Sketch）」と呼びます．

最初は円を描いてみましょう．スケッチ 1 の内容をエディタにすべて半角で打ち込んでください．入力ができたら，プログラムを実行するために▶のボタンをクリックし，結果を確認しましょう．

```
ellipse(50,50,80,80);
```

スケッチ 1　最初のスケッチ（円を描く）

小さなウィンドウが開き，そこに円が表示されたはずです（図 -6）．スケッチの実行を停止するには■のボタンをクリックします．命令の意味については，あとで詳しく説明していきます．エラーが表示されてうまく動作しないときは，スケッチ 1 の内容が間違いなく半角で入力できているかを確認してください．

図 -6　最初のスケッチの実行

新しいスケッチには「sketch_XXXXXXX」という名前が自動的につきます．XXXXXXX は 6 桁の日付と a からはじまるアルファベットです．スケッチを保存するには，ファイルメニューから行います．Windows の場合は，ドキュメントの Processing というフォルダの中に，スケッチごとにフォルダが作成されて保存されます．たとえば，2019 年の 10 月 28 日に最初に作ったスケッチは「sketch_191028a」というフォルダの中に保存されます．フォルダの中にある「sketch_191028a.pde」というファイルをダブルクリックすると，編集の続きが行えます．スケッチの名前は自由に変更することができますが，全角の文字は自動で「_（アンダーバー）」に置換されます．半角で分かりやすい名前を入力して保存するようにしましょう．

基本的な命令の形

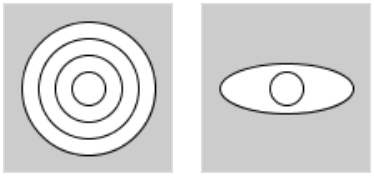
最初に試したスケッチ 1 の内容を詳しく解説し，命令の基本的な形やエラーについて整理しておきます．

スケッチ 1 は「左から 50 ピクセル，上から 50 ピクセルの位置を中心にして，幅 80 ピクセル，高さ 80 ピクセルの円を描け」という命令です．人間が相手の場合，紙を渡して「円を描いて」と頼めば，ほどよい大きさの円を描いてくれると思いますが，コンピュータの場合はそうはいきません．どの位置にどのような大きさの円を描くのか，細かく指示してあげる必要があります．

❑ 練習問題

スケッチ 1 に円を描く命令を追加して，お手本を参考に的と目玉の模様を描いてみましょう．

ヒント：上に書いた命令から先に実行され，画面の奥から図形が描かれます．



＜解説＞

的の模様を描くスケッチの例

```
ellipse(50,50,80,80);
ellipse(50,50,60,60);
ellipse(50,50,40,40);
ellipse(50,50,20,20);
```

目玉の模様の解答例は，スケッチ 3 を参照してください．

命令の基本形を図 -7 に整理しました．これを理解すれば，ほかの図形の描画などにも応用できます．Processing にはさまざまな「関数」と呼ばれる部品が用意されています．それらのうちの 1 つが円を描くための ellipse 関数です．関数は「よく使う便利な命令の集まりに，分かりやすい名前をつけて使えるようにしたもの」と理解すればよいでしょう．関数の処理に必要なパラメータ（動作を決める数値や文字など）をカッコで囲んで指定します．円の場合は 4 つのパラメータを指定して，描きたい円の詳細をコンピュータに伝えます．

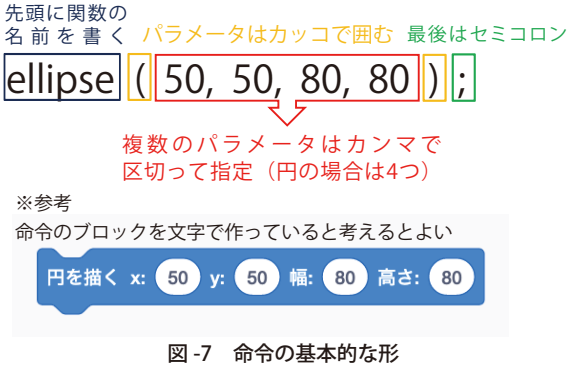


図 -7　命令の基本的な形

命令を書く際には，大文字小文字も含めて厳密に一字一句間違えないようにする必要がありますが，打ち間違いや勘違いは誰にでもあるものです．入力をしている途中や，スケッチを実行しようと思ったときに赤い波線が表示されたり，行が黄色でハイライトされたりすることがあります．これはスケッチの文法間違いなどのエラーを示しています．エディタの下のほうにあるメッセージエリアに，エラーの内容が表示されます．それをヒントに修正をする必要があります(図 -8)．

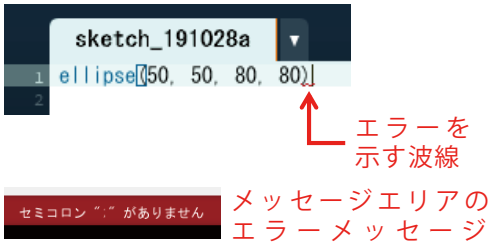


図 -8　エラーの表示

❑ 練習問題

次のスケッチはエラーで実行できません．修正すべき部分を考えてみましょう．実際にエディタに入力してみるのもよいでしょう．

```
ellipse 50,50,100,100;
ellipse(50 50,80,80);
elipse(50,50,60,60)
```

＜解説＞

1 行目：パラメータを囲むカッコがない
2 行目：最初の 50 の後にカンマが抜けている
3 行目：ellipse のつづりが間違っているし，行末のセミコロンがない

スケッチの命令はすべて半角で入力します．誌面だと分かりにくいですが，スケッチ 2 の 1 行目は最後のセミコロンが全角です．2 行目は 2 番目のパラメータの前に全角のスペースが入っています．スケッチ 2 のように書いてしまうと，エラーが発生して実行ができません．

```
ellipse(50,50,80,80);
ellipse(50,　50,80,80);
```

スケッチ 2　全角の入力でエラーになる例

命令の本体を書くのに全角は使えませんが，スケッチに日本語でメモ書きを加えることはできます．これを「コメント」と呼びます．複数行の場合は /* と */ で囲み，1 行の場合は // をはじめの部分に記入します．コメントは灰色の文字色で表示されます．目玉の模様を描くスケッチにコメントを追加した例がスケッチ 3 です．

```
ellipse(50,50,80,30); //白目を描く
ellipse(50,50,20,20); //黒目を描く
```

スケッチ 3　目玉の模様を描く(コメント付)

Processing のエディタにはスケッチを自動的に読みやすくしてくれる機能があります．編集メニューの「自動フォーマット」がそれです (図 -9)．こまめに自動フォーマットをすれば，カンマの後にスペースが挿入されたりして，スケッチを読みやすい状態に保つことができます．

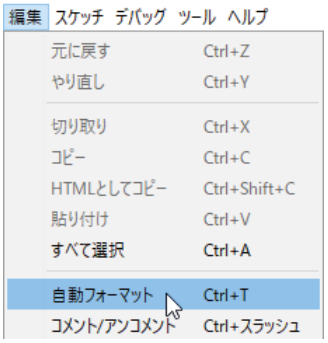


図 -9　自動フォーマット

座標と図形

コンピュータの画面はピクセルと呼ばれる細かい点が集まってできています．この点に座標をつけることで，画面上の位置を指定します．Processingの場合は，実行結果を表示する画面の左上が原点(0, 0)に設定されます．画面の大きさを指定しない場合は，自動的に縦横が100ピクセルの画面が開きます．スケッチ1を実行したときの画面の大きさと座標を図-10に示します．

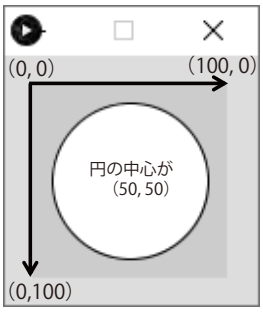


図-10　画面の大きさと座標

画面の大きさを指定したい場合は，スケッチの最初にsize関数による設定を加えます．幅480，高さ120のウィンドウを開き，ウィンドウの中心(240, 60)の座標に1ピクセルの大きさの点(point)を描くには，スケッチ4のように命令します．

```
size(480,120);
point(240,60);
```

スケッチ4　ウィンドウの中心に点を描く

基本的な図形を描画するための関数と座標などのパラメータを図-11に整理しました．試しにいろいろな図形を描いてみましょう．

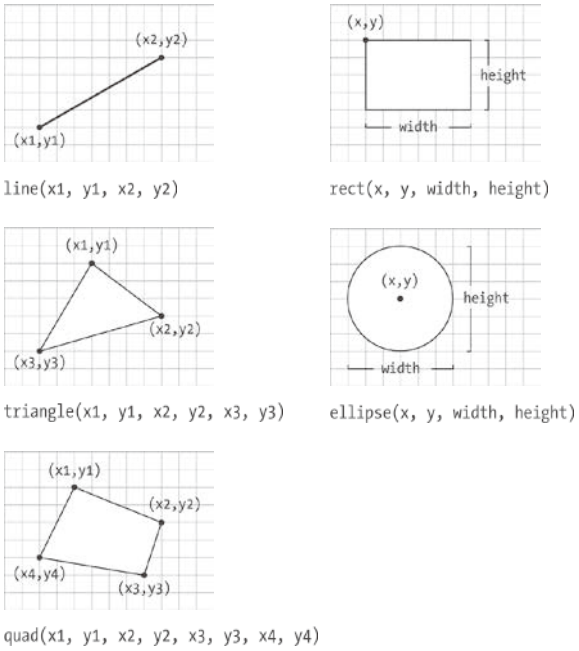
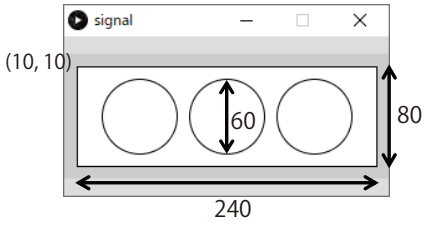


図-11　いろいろな図形を描く関数（文献1）P.19より）

● 宿題

お手本を参考に，信号機を描いてみましょう．今回はモノクロですが，次号で色を付けていく予定です．



画面のサイズは横260，縦100
円の中心は左から(60, 50)(130, 50)(200, 50)

参考文献

1）　Reas, C., Fry, B. 著，船田　巧 訳：Processing をはじめよう 第2版，オライリージャパン(2016)．

(2019年10月31日受付)

杉浦　学（正会員）　manabu@kamakura-u.ac.jp

鎌倉女子大学家政学部家政保健学科准教授．慶應義塾大学大学院政策・メディア研究科後期博士課程修了．博士（政策・メディア）．プログラミング教育をはじめとした情報教育に関する研究に取り組む．中高生向けの著書に『Scratch ではじめよう！ プログラミング入門 Scratch3.0 版』（日経 BP 社）など．