べた語義

Vol. 104

CONTENTS

COLUMN

携わるシステムが利用される楽しみ

実世界の社会インフラとして，橋を架けたとか，道を通したなど，それぞれの仕事を成し遂げた方々は，それらの利用状況を見て，自らが成したモノに対して，陰に陽に誇りに思うことがあると想像します．情報学関係の分野では，たとえば研究者なら提案手法やアルゴリズムといった研究成果が，ほかの研究の糧や礎になることもあれば，教育者ならば自らが教育文脈で関係した人の成長を見ることも類すると思います．

情報基盤や情報システムの場合を想像してみます．そうすると，たとえば内製の場合では，システム導入当初，利用者から聞こえてくる声の中には，“今までと違う（戸惑い）”，“使いにくい（批判）”といったネガティブなものもあります．そんなときには，提供側は，それらに対して“システムや基盤は常に進化・改善されます”とか“手順書は読みましたか？”，のような心の声を発することもあるかと思います．状況を否定的に捉えず，“ああ，使ってもらえている”とか“なるほど，〇〇にはまだ改善の余地がある”などと肯定的な姿勢であれば，精神面でも健全です．外注システムの場合でも，要件定義や外部仕様への関与，あるいは運用への寄与など携わり方はさまざまですが，直接的でなくとも，ある種の冷静で真剣な思いがより良いシステムに繋がることはよくあります．

否定的な声は的を射ていることも多いので，ありがたくひとつひとつの声を冷静に分析しながら，システム改善に繋げていけば，次のシステムではその経験が活かされます．使われる楽しみを覚えるのは，どの程度工夫し，どの程度魂を込めた（ベストを尽くした）か次第な部分があります．技術的に工夫した部分が「はまる」と喜びは倍増です．工夫の中には，簡単なスクリプトの提供だけでも現場からは大変重宝されることもあります．ただし，ライフサイクル上の契機により終息するシステムもあり，終息が決まったときにはスパッと気持ちを切り替えられる素養も大切で，それには経験が必要です．

実世界の道や橋にも大小あって，その利用者も地域の方だったり世界中の方だったりするわけですが，情報システムでも同様です．情報システムの文脈で何等かの対岸に橋を架け，道を通すことでさまざまな方に貢献できれば，この世界で生きていくひとつの動機（楽しみ）になり得ます．もしこれからシステム提供側を目指すなら，さまざまな声を冷静に捉えて「めげずに楽しむ」ための経験と自信を持ちたいものです．

松浦健二（徳島大学）

LOGOTYPE DESIGN...Megumi Nakata, ILLUSTRATION&PAGE LAYOUT DESIGN...Miyu Kuno

ARTICLE

ワークショップで小学生のための情報科学の授業を作った話

原田康徳
デジタルポケット

きっかけ

とうとう小学校でのプログラミング教育が始まってしまいました．と言っても，みなさんご承知の通り，プログラミングを学ぶ科目ができたのではなく「プログラミング的思考」を身につけさせるという，なんとも不思議な内容になっています．これまでにもコンピュータを操作する技術はいろいろと教えられてきましたが，コンピュータの中身に近づいたのは評価すべきだと思います．しかし内容が，30年くらい前のコンピュータ観で止まっていて，たとえば，順次，繰り返し，条件分岐がプログラミングの基礎というような授業が行われていたりします．これは間違いではないですが，制御構造だけ取り出しても意味はなく，変数とセットで教える必要があります．ところが変数を教えるのは難しいので，ロボットや亀に変数の代わりをさせますが，そもそも，制御構造だけ教えることってそんなに大事なの？と思うわけです．

コンピュータの技術は目まぐるしく変化しているなかで，小学生への教育を考えることはとても難しいことです．古いことは必ずしも基礎ではないですし，今の時代の最先端が，子どもたちが大人になるころでも最先端である可能性もほぼありません．専門家を育成する教育だったら全部教えればよいのですが，義務教育として市民全員が知らなければならないことはどんなことなのか．その切り口を考えるのは非常に難しい．

そのためには，コンピュータの技術の変化を流れとして見る必要があります．コンピュータの専門家たちはコンピュータをどのように考えているのでしょうか．ここで僕が言っているコンピュータの専門家というのは，コンピュータに関する技術を前に進める論文を書いたことのある人で，単にプログラミングを職業としている専門家ではありません．

そんなモヤモヤを吹き飛ばしてくれるチャンスがやってきました．僕に情報処理学会「夏のプログラミング・シンポジウム」の幹事をやらないかと．テーマは子どもに向けたコンピュータ教育．それで今までやってみたかった，コンピュータの専門家に子ども向けの授業を作ってもらうことにトライしました．

「課外授業　ようこそ先輩」というNHKの番組がありましたが，各界の有名人が母校に帰って良い授業をするという，あんなイメージです．授業はその分野の本質に迫っていて，大人が見ても感心するような内容です．

さあ，それを夏のプロシンのフォーマットに合わせてどうインプリするか．やったことのないことですから慎重になります．幹事団の議論は非常に盛り上がりました．

最初の山場は，参加者が集まるかどうか．会場費と参加費から赤字にならない最低参加人数は決まりますが，不安だらけなので対象を広め広めで募集しました．で，ありがたいことに結構な人数の参加と発表申し込みをいただきまして，そこから具体的な進行を考えるフェーズに入りました．

事前準備

最終的なゴールはモデル授業をいくつか作ること．グループに分かれて授業を作って最後にその授業の体験をして終わることにします．いままで思いついたことのない授業を作るという，とても創造的な活動になるはずです．それで，効果の程は分かりませんが，全体をワークショップ形式で作ってゆくことにしました．

僕は以前はワークショップを見様見真似でやってましたが，ちゃんと基礎から学びたいと思い，10年前に青山学院大学の「ワークショップデザイナー育成プログラム」というのを受講しました．期間は3カ月ですが，ワークショップの基礎理論を学んで，実際にグループでワークショップを作ってそれを子どもたちにやってみるという内容です．今回はそこでの流れをかなり参考にしています．僕に加えて幹事の渡辺勇士さんもワークショップデザイナーですが，さらにグループに一人ずつそれが分かっている人に入ってもらいたい．でも仕事として頼むほどの余裕はない．ということで，普段ビスケットのワークショップをやってくださっている方々でワークショップデザイナーの講座を修了した人たちにお手伝いに来ていただくことにしました．

インプット

例年の夏のプログラミングシンポジウムと同様に一般発表を募集しまして，それに加えて，模擬授業と基調講演もお願いしまして，参加者へのインプットとします．ざっくりとした会の時間割は，初日と2日目の午前中で，一般発表6件，模擬授業2件，基調講演，2日目の午後からグループで授業を作り始めて3日目の午前中に作った授業の体験をやって終わりという流れです．

模擬授業の1つは長崎県立大学の山口文彦先生による高校生向け「暗号の話」出張授業です．1つはアルファベットが書かれたテープを2本用意して，鍵となる数だけずらして文字を置き換えるシーザ暗号の体験．もう1つは，表に公開する2つの数，裏に秘密の数1つが書かれたカードを使い，安全に暗号の鍵を伝える体験．なるほど，暗号もこういうやり方だったら分かりやすい．やり方を工夫すれば小学生にもできそうな内容でした．

もう1つの模擬授業は東北大学の中野圭介先生による「モデル検査のパズル化」です．列車の車両の部品があり，切り口の形で連結できるかどうかが決められている．両端が決まっているため，一定のルールでしか完成させることはできないが，遊びながらどんなルールが隠されているのかを探ってゆく（右の車両はパンダを1，カエルを0とする2進表記で上の列を3倍した数が下の列に現れるそうである）．実際に小学生にやった内容だそうで，簡単そうに見えて意外と奥が深くて面白いものでした．この2つの授業のおかげで，参加者の皆さんにイメージが湧きやすくなったように思います（図-1）．

電気通信大学の久野靖先生には，いま先生が日本学術会議の中で策定されている「情報教育の参照基準」についての基調講演をお願いしました．大学生・高校生・中学生・小学生と各年代で非常に広い範囲で学んでほしい情報教育を網羅的に整理した基準の紹介です．そういうしっかりしたものがあるおかげで全体のバランスを気にせずに，参加者は羽目を外した内容に挑戦できるようになるだろう．という意図がありました．

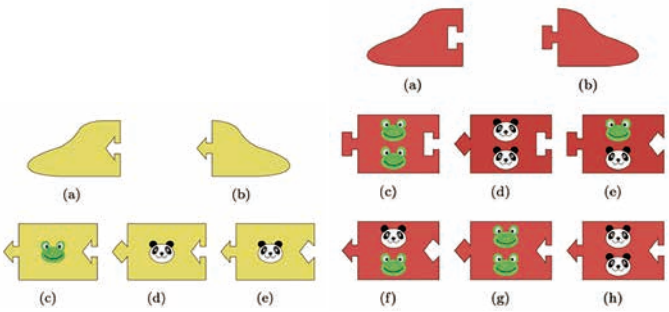


図-1　車両のパズル

グループ決め

参加者は若い学生さん，リタイヤされた大学の先生，バリバリの研究者，コンピュータを使えるレベルの素人の方と多岐にわたりました．それらからどうやってグループを作るかです．グループワークがうまく回らず，途中で喧嘩が起きてしまうのは論外ですが，専門家が頭の中で考えたものじゃなくて，ちゃんと素人に伝わる内容になっているかも大事です．つまり，グループの中に専門家と「子どもの視点に立てて」「適度に素人目線」な人が混ざっている必要があります．さらに，運営側でエイやと決めてしまうのではなく，自由に好きな人と一緒になってでもなく，なんらかの興味の一致する人たちが緩い必然性でグループができて．さらにグループ内で上下関係がなくフラットな関係が維持できるような人間関係．グループは 2 日目の午後までに決めたいので，その準備として初日の夜が使えます．通常なら懇親会で楽しくお話をする時間ですが，それをワークショップ的に何かできないか，と考えました．

それで，行ったワークショップは次のようなものです．

1) 自己紹介〜コンピュータ自分史〜

A3 の紙に 4 つの質問について簡単に答え，それを持って，4 人グループで自己紹介を 3 セット．質問内容は，子どものころ好きだった遊び，コンピュータとの出会い，衝撃を受けたコンピュータのエピソード，今の自分（専門分野）ということで，自分が子どもの頃を思い出しながら，どうして今の自分があるかを振り返ってもらいました．

2) コンピュータとは〇〇

自分が考えるコンピュータを一言で言い表してもらって，同じようにグループで話し合う時間．自分史からさらに進んでコンピュータに対するイメージを共有します．専門家が考えているイメージと，素人が考えているイメージの違いなどにも気づいてもらいます．

3)「コンピュータとは」地図づくり

グループに分かれて模造紙上で，全員が考えたキーワードの地図を作りました．この作業が 1 つのグループワークですが，同じキーワードの集合なのにグループごとで分類の仕方が全然異なるなど面白い結果になりました．

4) モヤワード

1) から 3) のワークを通じて，意味の分からない言葉，気になった言葉を「モヤワード」としてメモをしておき最後に回収しました．

この話題と進行は，さまざまなレベルの参加者が対等に語り合え，専門家があえてこだわりたくなるようなものを選びました（図 -2）．

2 日目の午後のグループ分けは，2 段階で行いました．まずは，6 グループがそれぞれ作った「コンピュータとは」の地図，モヤワード，個人的に提案されたテーマ（3 つ）から，一人 3 票の投票で候補を 6 つに絞ります．そこから，自分が参加したいグループを自由に選びます．人数の制約は各グループ 3 〜 6 名です．

これによって決まったグループは次のようになりました．

1. モヤワード「コンピュータに教える」
2. モヤワード「コンピュータの面白さが分からない」

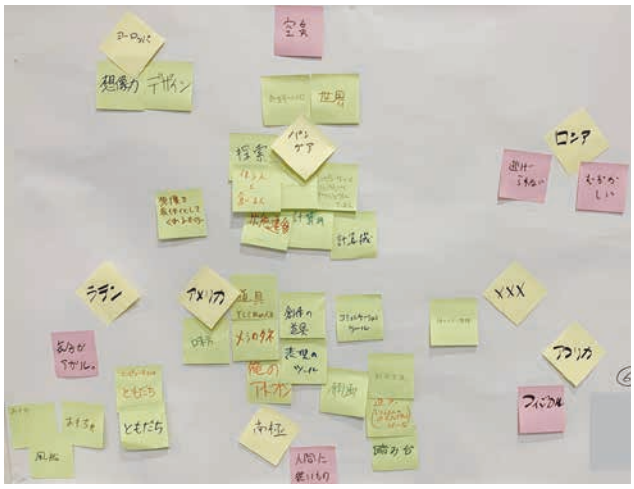


図 -2 「コンピュータとは」の地図　各自が思っている「コンピュータとは〇〇」をグループごとに 2 次元に配置した例．このグループは世界地図に喩えてユニークに配置している

3. 地図「パンゲア」
4. 提案テーマ「先生にやりたいと思ってほしい授業」
5. 提案テーマ「playful で試行錯誤ができる講義から入らない小学生向けの授業とは？」
6. 提案テーマ「コンピュータサイエンスを国語で考える」

授業作り開始

ここから皆さんに作っていただく課題は次のような制約でお願いしました．

目的：コンピュータを知ってもらう
対象：小学生（中学生でも可）
時間：45 分
場所：学校の教室
伝えたいこと：グループで考える
内容（体験と解説）：グループで考える

2 日目の午後以降は，グループ単位での授業作りの時間としました．この時間はみなさんとても真剣に議論されていたようです．最終日に 6 グループの授業を 2 つずつ同時に実施し 3 セット行う予定でしたが，2 日目で大体できてしまった 2 グループには夜に前倒しで実施してもらいました．いずれの授業も短時間で完成度の高いものを作っていただき，ほかのグループへの良い刺激にもなったようです．予想されていましたが，消灯時間 12 時まで（お酒も飲まずに）作業をしていたグループもありました．

最終日の授業も，本当に幼稚園の手遊び歌から始めたグループも現れるなど，楽しい内容のものばかりでした．笑い声がおき真剣に取り組む様子が見られました．

授業をやって終わりではなくて，ここから改善の時間をとりました．そのために，授業ごとに参加者からフィードバックをもらいインタビューをしました．このやり方も，実際にワークショップデザイナーの講座で行われている手法をほぼそのまま使わせていただいてます．

参加者には，改善のアドバイスではなく感想のみを付箋紙に書いてもらいます．

赤い付箋紙：面白かった・夢中になれたところ
青い付箋紙：やりづらかった・違和感を覚えたところ

我々は「面白かった」という感想の前に，つい改善点のアドバイスを言ってしまいがちですよね．まずは，ポジティブな感想で，このオリジナルな授業を作ったことを称えましょう．ときには本当に重要なアドバイスが言えることがあるのかもしれませんが，しかし大抵の場合は「やりにくかった」という，実施している人たちが見えていない視点を伝えるだけで十分なのです．改善点は自分たちで見つけ出せるはずですし，その方が作った人たちの気分が良いですよね．

最後にまた 40 分ほどグループでフィードバックを受けて修正をし，グループごとに発表をして会は終了しました．

結果報告

できあがった授業をご紹介します．紙面の都合上，詳しくお伝えするのは最初の 2 つだけです．

●コンピュータの入力について勉強しよう

チーム：1-2-5 木
ねらい：テーマ「コンピュータに教える」

情報を入力するとはどういうことか？について AD 変換／量子化の動作を体感してもらう
内容：同じ大きさの板を 5 枚ずつ重ねたもの，2 枚ずつ重ねたもの，1 枚ずつを複数個用意して，それをものさしとして長さを測り，相手に数で伝える．聞いた側はその数から長さを再現する．5 のものさしだけを使った場合，2 のものさしだけを使った場合，1 のものさしだけを使った場合で，測る長さの正確さが違うことを体験する（図 -3）．
解説：コンピュータにはものさしで計ったようにしか入力できない．エアコンは部屋の温度を計って動

くけれど，部屋の温度をどれくらい細かく計るのか，5のものさしのように飛び飛びで計るとちょうどいい温度で調整するのがむずかしくなる．

●コンピュータって面白いの?

チーム：フェイスホワイト

ねらい：コンピュータを知ってもらう

コンピュータは正確に動く

コンピュータはアホなことでも命令されたらやる

コンピュータの面白さは，自分が作る

コンピュータって面白いの？，面白さが分からない，を解消するために作った授業

内容：「start」ボタン1つと文字が入力できるマスが2×6行ある画面の動作を隠し自由に触らせ動きを探る．種明かしは「start」を押すと右のマスの文字がすべて消え，左のマスの数秒後，右のマスの文字を再表示するというもの．表示させる文字やタイミングを工夫して「何か面白いことをしてください」と指示し自由制作（図-4）．

解説：コンピュータは「正しいことしかしない」ではなく，「正しくないことも命令すれば，全力でやってくれる」．

「コンピュータが面白い」のではなく「コンピュータ」は命令を順番にやるだけ．

図-3　積木を重ねて作ったものさし．この厚さを単位として測る

「面白い」ことを考えるのは人間の役目だ．

●コンピュータとのかかわりを見つけよう

チーム：パンゲア

内容：「電気にかかわるもの・繋がるもの」「そうでないもの」の写真をたくさん撮り，それらの写真を「入っている」「おなじやくめ」といった接続するコマでつないでゆく対戦型ゲームで進める．IoTの時代を見据えて，身の回りのものとコンピュータとのかかわりを考えてゆくきっかけとする．

●コンピュータの気持ちになろう

チーム：ぱわぽん

内容：命令カード：数字，「に移動」，「下に書く」，「右に書く」を並べてプログラムが作られている．それを解釈し実行すると絵がかける．用意されたプログラムを人間が実行して絵（漢字）を書いてみたり，与えられた漢字を描くプログラムを作ってみる．命令とデータ，抽象化，コンピュータは不満を言わない，といったことを学ぶ．

●意外と伝わらない!?「伝言ゲーム」

チーム：うさぎさんチーム

内容：用意された絵（「イ」の逆さまのような）を数字を使わずに相手に伝えて，書いてもらう遊びを通じて，曖昧な指示の伝わりにくさ（コンピュータは数字の指示が得意），言葉を節約する方法を通じて抽

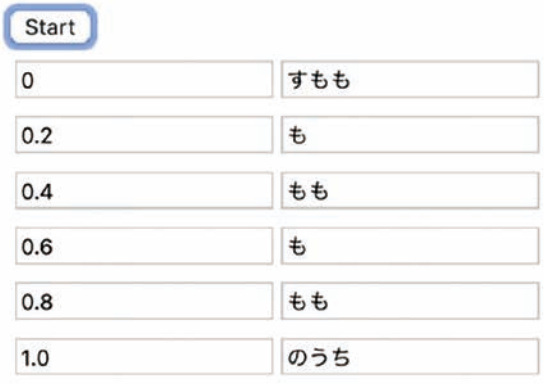


図-4　画面の例

象化を学ぶ．また，できあがった絵が錯覚になっている二重の驚き．

●観察して・まぜて・面白くする国語

チーム：いいができた

内容：「タオルを」「首に巻いている」といった文の主語と述語を交換し，意味の通らない文を作る．意味が通るように間に入る言葉を考える．国語とコンピュータの面白さを伝える授業．

これからどうする？

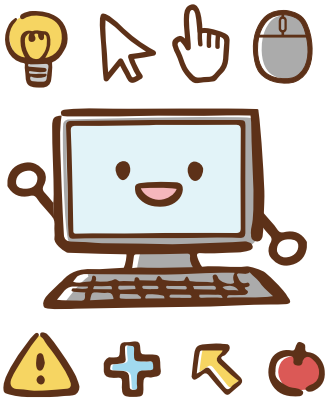
元々子どもへの教育に興味のある人たちが集まったこともあって，皆さんはとても楽しそうに授業作りにかかわってくださいました．どの授業もオリジナリティに溢れるものとなったと思います．次の課題は，これをどのように次の活動につなげるかです．実際に本物の小学生相手にやってみるとか，できた授業を本にしたらどうかといったアイディアもありましたが，実現には至ってません．

いずれにしても，元々は小学校への教育にもっと骨太なコンピュータを伝えたいというところにありました．10 年後の次の学習指導要領の改定まで，ゆるゆると進めて行ければとも思います．これを読まれたみなさんも，授業を作ってみたくなりませんでしたか？　ぜひ作ってみて，いろいろな形で広めていきましょう．

（2020 年 2 月 13 日受付）

原田康徳　hakase@viscuit.com

ビスケット開発者．博士（工学）．ワークショップデザイナー．1963 年北海道生まれ．1992 年北海道大学大学院情報工学専攻博士後期課程修了．1992 ～ 2015 年日本電信電話（株）NTT 基礎研究所，NTT コミュニケーション科学基礎研究所 1998 ～ 2001 年 JST さきがけ研究員．2004 ～ 2006 年，2010 ～ 2013 年　IPA 未踏ソフトウェア創造事業プロジェクトマネージャ兼務．NTT を退職後，合同会社デジタルポケット設立．



ARTICLE

Processing でプログラミングに挑戦！ —第 4 回 アニメーションとインタラクション—

杉浦　学
鎌倉女子大学

前号の宿題

前号では，カラフルな円の模様（図 -1）を描く宿題を出題しました．

● 前号の宿題

図 -1　カラフルな円の模様

この宿題の解答例をスケッチ 1 に示します．7 行目から 12 行目では，前号で解説した「繰り返し」と「乱数」を利用しています．繰り返しの for ループを二重にすることで，横一列に円を描く（内側のループ）ことを，画面の上から下まで（外側のループ）行っています．

```
1  //描画の準備
2  size(480,480);
3  background(0);
4  noStroke();
5
6  //カラフルな円を敷き詰める
7  for(int y=0; y<=height; y+=40){
8    for(int x=0; x<=width; x+=40){
9      fill(random(256),random(256),random(256));
10     ellipse(x,y,35,35);
11   }
12 }
```

スケッチ 1　カラフルな円の模様を描く（宿題の解答例）

二重の for ループの部分について，詳しく解説しておきます．内側の for ループ（8 行目から 11 行目）で 40 ずつ x を増やすことを繰り返し，さらにそれを外側の for ループ（7 行目から 12 行目）で 40 ずつ y を増やしながら繰り返します．最初に，外側の for ループで y が 0 からスタートし，内側の for ループで x が 0 から width（今回は 480）まで 40 ずつ増えていきます．具体的には，x が 0 → 40 → 80 → 120 →（中略）→ 400 → 440 → 480 と増えていくことで合計 13 個の円を描くことになります．この間は y の値は 0 なので，ウィンドウの一番上（図 -2 の上）の位置に円を描くことになります．次に y を 40 増やし，同じように x を 0 から 480 まで増やしながら 13 個の円を描きます（図 -2 の下）．このような動作を行うことで，ウィンドウの一面に円が敷き詰められていきます．

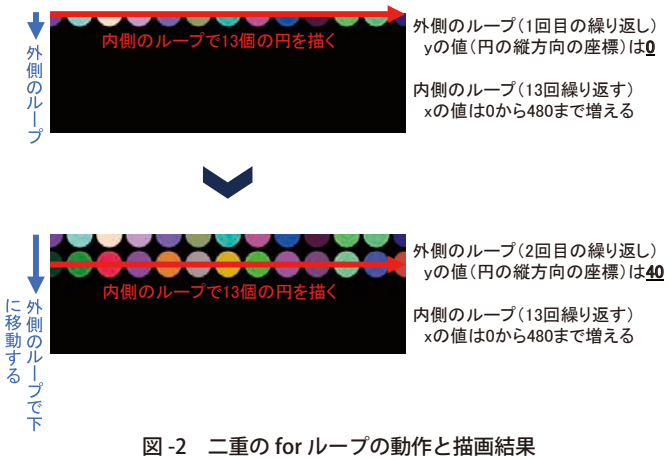


図 -2　二重の for ループの動作と描画結果

```
1  int x=0;
2
3  void setup(){
4    size(480,120);
5    noStroke();
6  }
7
8  void draw(){
9    background(204);//残像を消す
10   ellipse(x,60,9,9);
11   x++;//円を右に少し動かす
12 }
```

スケッチ 2　移動する円のアニメーション

図 -3　スケッチ 2 の実行結果

アニメーション

ここからは Processing でアニメーションを表現する方法について解説していきます．これまで作成してきたスケッチは，1 行目から順番に処理が実行され，最後の行に到達すると実行が終了していました．

Processing では，スケッチ 2 のように記述することにより，アニメーションを作ることができます．スケッチ 2 を実行すると，白い円がウィンドウの左から右に移動していくアニメーションが表示されます（図 -3）．スケッチ 2 では，1 行目で x という変数を宣言し，0 を代入しています．これは円の横方向の位置を保存しておくための変数です．3 行目から 6 行目の void から始まる部分と，同じく 8 行目から 12 行目の void から始まる記述は「関数」と呼ばれる部品です．void という記述以外で関数を書き始める場合もありますが，ここでは説明を省略します．void の後に半角のスペースを入れて，関数の名前を書きます．アニメーションを作る場合は「setup」と「draw」という名前にしておく決まりです．名前の後には「(」と「)」を書きます．今回は空の括弧ですが，括弧の中に何かを書く場合もあります．次に「{」と「}」で関数の範囲を指定します．

setup 関数と draw 関数を用意したスケッチは，図 -4 に示した順序で実行されます．スケッチ 2 の場合は，最初に 1 行目の変数 x の宣言が実行された後に，setup 関数の「{」と「}」に囲まれた部分が 1 回だけ実行されます．ここではウィンドウの大きさと輪郭線なしの初期設定をしています．次に draw 関数の「{」と「}」に囲まれた部分が，スケッチの実行が終了されるまで（■のボタンが押されるまで），1 秒間につき 60 回繰り返して実行されます．スケッチ 2 の場合は，画面全体を一度塗りつぶし，変数 x を x 座標にした位置に円を描き，x を 1 増やすという 3 つの処理が繰り返されることになります．結果として，円が左から右に動くアニメーションが実現できます．

図 -4　アニメーションを表示するスケッチの仕組み

描画の結果はウィンドウに残るので，円が移動しているように見せるために，9 行目でウィンドウの全面を灰色に塗りつぶしています．この塗りつぶしの処理がないと，円が移動していくたびに前回の描画の結果が残るので，図 -5 のように白い棒が伸びていくようなアニメーションが表示されます．

図 -5　円が重なって描画される

❑ 練習問題

前回の宿題の解答例（スケッチ 1）をアニメーションに改造してください．スケッチを実行するとさまざまな色の円がウィンドウに表示されるようにしてみましょう．draw 関数が 1 秒間に実行される回数（フレームレートと呼びます）を 5 回に設定するために，setup 関数に frameRate(5); を追加しましょう．

＜解説＞

前号の宿題をアニメーションさせるスケッチの例

```
1  void setup(){
2    size(480,480);
3    background(0);
4    noStroke();
5    frameRate(5);//1秒間にdraw関数を5回実行
6  }
7
```

```
8  void draw(){
9    background(0);//前のフレームの描画結果を消す
10   for(int y=0; y<=height; y+=40){
11     for(int x=0; x<=width; x+=40){
12       fill(random(256),random(256),random(256));
13       ellipse(x,y,35,35);
14     }
15   }
16 }
```

インタラクション 1　マウスの座標

アニメーションの方法が分かったところで，次はマウスやキーボードなどのユーザの入力に反応する「インタラクション（対話的なやりとり）」について解説していきます．

スケッチ 3 のようにすることで，マウスの位置に円が描けるようになります．実行をしてウィンドウの中でマウスを動かした後の様子を図 -6 に示しました．3 行目の fill 関数は 2 つのパラメータをとっていますが，2 番目は透明度で 0（完全に透明）から 255（不透明）の値で指定します．これにより，少し透けた黒色の円を描くように設定しています．8 行目の mouseX と mouseY は，現在のマウスの位置の座標が格納されている変数です．これらの変数は，height や width と同じように宣言をしないで使える変数のうちの 1 つです．8 行目でマウスの位置を中心にして円を描くように指定しています．マウスを速く動かした場合は円の重なりが少なくなって薄い黒色に，ゆっくり動かすと重なる円の数が増えるため濃い黒色になります．

```
1  void setup(){
2    size(480,120);
3    fill(0, 102);
4    noStroke();
5  }
6
7  void draw(){
8    ellipse(mouseX, mouseY, 9, 9);
9  }
```

スケッチ 3　マウスの位置に円が描かれる
（文献 1）p.58 より）

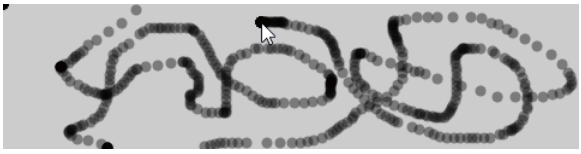


図 -6　スケッチ 3 の実行結果

インタラクション 2　マウスクリック

マウスの位置だけでなく，マウスボタンの状態を調べることもできます．マウスのボタンが押されると，mousePressed という変数の値が変化します．この変数の型はブーリアン型と呼ばれており，変数の値は真（true）か偽（false）のどちらか一方です．マウスのボタンが押されている間は，mousePressed の値は真（true）となり，押されていない場合は偽（false）となります．

この変数の値を調べることで，マウスがクリックされたときに特定の処理を実行することができます．まずはこの部分について詳しく考えていきましょう．スケッチ 4 では，マウスのボタンがクリックされたときに線の色が変化（図 -7）します．

```
1  void setup(){
2    size(240,120);
3    strokeWeight(30);
4  }
5
6  void draw(){
7    background(204);
8    stroke(102);
9    line(40,0,70,height);
10
11   // マウスがクリックされていたら線を黒に
12   if (mousePressed==true) {
13     stroke(0);
14   }
15
16   line(0,70,width,50);
17 }
```

スケッチ 4　マウスのクリックに反応する
（文献 1）p.64 より）

図 -7　スケッチ 4 の実行結果

ある条件が成立しているかを調べて，成立しているときにだけ指定した処理を実行したい場合は，スケッチ 4 の 12 行目から 14 行目のように if を使った「条件分岐」を記述します（図 -8）．条件分岐は繰り返しの for ループと似ています．括弧中の条件の記述は，前号で紹介した比較演算子を使って記述します．

```
if (条件) {
  条件が成立したときに実行されるコード
}
```

スケッチでの書き方

```
if (mousePressed) {
  stroke(0);
}
```

条件に記述する変数の値が真偽の場合は == true を省略できる

参考：ブロック型の言語では

図 -8　条件分岐の記述方法

ある条件が成立したときと，それ以外のときに別々の処理を実行したい場合は，if の「{」と「}」の後に else を追加します．スケッチ 4 の条件分岐の部分（12 行目から 14 行目）を，else を使ったものに変更したものをスケッチ 5 に示します．実行結果と動作の解説は図 -9 をご覧ください．

```
if(mousePressed){
  stroke(0);
}else{
  stroke(255);
}
```

スケッチ 5　else を追加したスケッチ 4 の抜粋
（文献 1）p.65 より）

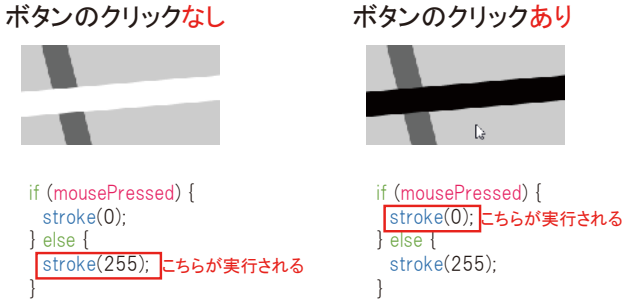


図 -9　スケッチ 5 の動作と解説

● 練習問題

最後の練習問題として，簡単なアート作品を作成してみましょう．以下に示したスケッチは，毎回同じ位置に，1 秒間に 1 回ずつ，半透明の赤い円が繰り返して描かれます．

```
1  void setup(){
2    size(800,600);
3    background(0);
4    noStroke();
5    frameRate(1);
6  }
7
8  void draw(){
9    fill(255,0,0,90);//90は透明度を指定
10   ellipse(400,300,30,30);
11 }
```

このスケッチを，位置と色と大きさがランダムな円が増えていくようなアニメーションに変更してみましょう．

次のような順番で作業をしてみるとよいでしょう．

Step1：円の描画位置を毎回異なる値に変更しましょう．x 座標と y 座標に random 関数を使って乱数を指定しましょう（横は 0 から width，縦は 0 から height）．

Step2：円の色を赤で固定ではなく，いろいろな色に変更してみましょう．fill 関数の RGB のそれぞれの値を random 関数で置き換えます（0 から 255）．

Step3：円の大きさを 50 以上・200 未満の間でランダムになるようにしてみましょう．
注意：横幅と縦幅を一緒にしないと正円にならないので，生成した直径を変数に保存し，横幅と縦幅の両方で参照しましょう．
ランダムな直径を変数に保存するヒント：

```
float diameter = random(50,200);
```

Step4：フレームレートが少し遅いので，1 秒間に 10 フレームが実行されるように変更しましょう．

一通り完成したら，マウスをクリックしている間だけアニメーションが進んだり，マウスの位置に円が描かれたりといったアレンジをしてみるのもよいでしょう．

＜解説＞

Step4 まで作業をしたスケッチの例

```
1  void setup(){
2    size(800,600);
3    background(0);
4    noStroke();
5    frameRate(10);
6  }
7
8  void draw(){
9    fill(random(256),random(256),random(256),90);
10   float diameter = random(50,200);
11   ellipse(random(width),random(height),diameter,diameter);
12 }
```

投稿のすすめと発展学習

これまで 4 回にわたって，中高生のジュニア会員の皆様を読者に想定した連載を掲載しました．ページ数の都合から詳細の説明を省いた部分もありますが，Processing のようなプログラミング言語と，それを用いた創作活動に興味を持つきっかけとなれば幸いです．

また，ジュニア会員の皆さんがプログラミングに挑戦した結果は，ぜひ本誌の連載「集まれ！ジュニア会員！！」のページに投稿（投稿方法は https://www.ipsj.or.jp/magazine/jrlist.html を参照）してみましょう．たとえば，最後の練習問題を少しアレンジしたような，シンプルなスケッチも大歓迎です．

Processing はさまざまな創造的な表現をする仕組みが整っています．最初の一冊としては，文献 1) に示した『Processing をはじめよう』がおすすめですが，より発展的な内容も含めて学習したい場合は，文献 2) に示した『Processing クリエイティブ・コーディング入門』も手に取ってみてください．

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）．
2) 田所　淳：Processing クリエイティブ・コーディング入門―コードが生み出す創造表現，技術評論社（2017）．

（2020 年 1 月 4 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．