140 文字 Processing プログラミング

宮代 理弘[1]　宮下 芳明[1]

概要：本稿では，プログラミング言語である Processing のコードをできる限り短くしてツイートする遊びについて述べる．そこでは，「プログラムがツイートされると，それを見た人がさらに短いコードを作成してツイートする」という一種のコミュニケーションがみられる．実例をもとにそこで観測されたコード短縮技法やコミュニケーションを紹介する．

Programming in 140 characters with Processing

Miyashiro Masahiro[1]　Miyashita Homei[1]

Abstract: In this paper, we discuss a game to shorten and tweet source codes written in Processing. When one tweets a code, another user tweets a shorter one; we think of that tweets as a kind of communication. We illustrate the techniques of shortening and some type of communication by several examples.

1. はじめに

明治大学総合数理学部先端メディアサイエンス学科 [1] では，1 年次からのゼミ配属や Twitter の積極的な導入を特徴としたカリキュラムにより教育を行っている．1 年次の第 1 クォータ（4 月〜）で，HSP を用いたプログラミング自由創作体験「エンタテインメントプログラミング演習」を履修したのち，第 2 クォータ（6 月〜）から，Processing を用いたプログラミングの授業「プログラミング演習 I」が始まる．この授業では，オリンピックの五輪マークの描画や信号機の挙動の模倣といった課題を，時間内に解いて提出することが求められている．

ここで筆者らは，こうしたカリキュラムとは独立したエンタテインメントとして，すでに終了した課題プログラムを 140 字以内に短縮してツイートする試みを行った．ソースコードの短縮には，通常のプログラミングの授業では教わらない工夫が必要であった．さらに，ツイートされたプログラムに対して更なる短縮案が他のユーザからツイートされるという緩やかな競争・共創・狂想が起こった．

これに類する事例として，オンライン／オフラインにおける GitHub[2] やハッカソンでの共創，プログラミングコンテストやハッキングコンテスト [3] などの競技としての競争，普通ではないプログラムを披露し合う ABPro[4] での狂想（A と B の二文字によるプログラミング言語，Excel VBA による音楽シーケンサ，VGA 端子ハック）などがある．本稿での 140 字 Processing プログラミングは，これらのどれとも異なる面白さをもっており，従来のコードゴルフ [5] やショートコーディング [6] にはなかったコミュニケーション要素もみられる．

本稿では，実際にツイートされた事例をもとに，これらの工夫や競争・共創・狂想のさまを分析し考察する．

2. 事例紹介

この章では，2014 年 5 月以降の関連ツイートで筆者が確認できたものの中から特徴的な事例を紹介する．

2.1 五輪マーク

五輪マークを描画する課題（**図 1**）は，それ自体は大して難しいプログラムではない．しかし通常は，図 1（左）のように線の色を指定する stroke 関数と円を描画する ellipse 関数を各々の輪に対して記述する必要がある．そのため，140 文字でつぶやくためには工夫が必要となる．

図 2 は，五輪マークを生成するプログラムであり，出力

[1] 明治大学総合数理学部先端メディアサイエンス学科
Department of Frontier Media Science, Faculty of Interdisciplinary Mathematictical Sciences at Meiji University

```
noFill();
// blue
stroke(0,0,255);
ellipse(15,15,25,25);
// yellow
stroke(255,255,0);
ellipse(30,30,25,25);
// black
stroke(0,0,0);
ellipse(45,15,25,25);
// green
stroke(0,255,0);
ellipse(60,30,25,25);
// red
stroke(255,0,0);
ellipse(75,15,25,25);
```

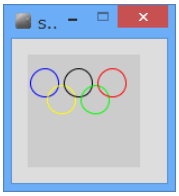
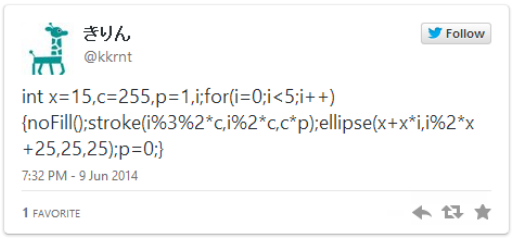


図 1　五輪マークを描画するコードと出力結果（コメント文を除き，194 文字）

Fig. 1　A code drawing Olympic mark and an output (194chars except comments).




```
int x=15, c=255, p=1, i;
for (i=0;i<5;i++) {
  noFill();
  stroke(i%3%2*c, i%2*c, c*p);
  ellipse(x+x*i, i%2*x+25, 25, 25);
  p=0;
}
```

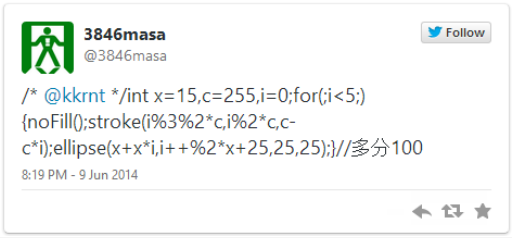
図 2　@kkrnt による五輪マークを描画するコード（108 文字）

Fig. 2　The code drawing Olympic mark tweeted by @kkrnt (108chars).

結果は図 1（右）の通りである．図 2 のコードは，複数回使う数値に関しては変数を使う，指定する数値を数式として一般化して for 文によって繰り返すといった手法を用いることでコードの短縮化を図っている．これにより 140 文字以内に収められ，Twitter 上で公開することができている．

図 2 のコードをもとにして，他のプログラマがさらに短く改良したものが**図 3** である．このコードは，for 文の更新部分に ellipse 関数の引数内に入れ込む，変数をひとつ減らすために数式を調整するなどの工夫を施し，コードの短縮に成功している．

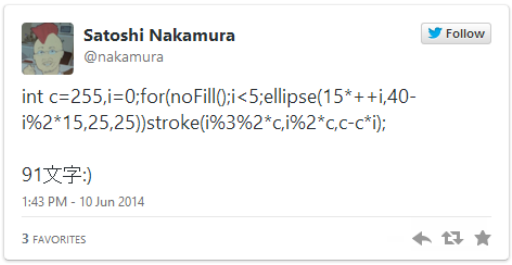
さらに，それらのコードを改良したコードが**図 4** である．このコードは，for 文の初期化部分と更新部分に関数を埋め込み for 文内のコードを 1 文にすることで，for 文の外側にある波括弧の省略が実現できている．




```
int x=15, c=255, i=0;
for (;i<5;) {
  noFill();
  stroke(i%3%2*c, i%2*c, c-c*i);
  ellipse(x+x*i, i++%2*x+25, 25, 25);
}
```

図 3　@3846masa による五輪マークを描画するコード（100 文字）

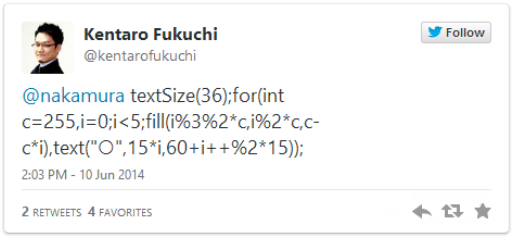
Fig. 3　The code drawing Olympic mark tweeted by @3846masa (100chars).




```
int c=255, i=0;
for(noFill();i<5;ellipse(15*++i,40-i%2*15,25,25))
  stroke(i%3%2*c, i%2*c, c-c*i);
```

図 4　@nakamura による五輪マークを描画するコード（91 文字）

Fig. 4　The code drawing Olympic mark tweeted by @nakamura (91chars).



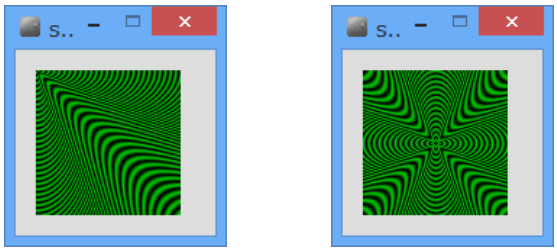

```
textSize(36);
for (int c=255,i=0;i<5;fill(i%3%2*c,i%2*c,c-c*i),text("
○",15*i,60+i++%2*15));
```

図 5　@kentarofukuchi による五輪マークを描画するコード（89 文字）

Fig. 5　The code drawing Olympic mark tweeted by @kentarofukuchi (89chars).

一方，他の方面からのアプローチで同じ文字数で実現したコードが**図 5** である．このコードは，円を描画する関数の代わりに，“○”の文字を描画する関数を利用している．また前述のコード同様，for 文の中に関数を入れ込むことでコードの短縮を行っている．

以上の流れからもわかるように，Twitter を通して不特定多数にコードを公開することによって，多くの人によっ



（1）　（2）

図 6　動的に変化するグラフィックス

Fig. 6　The dynamic graphics.

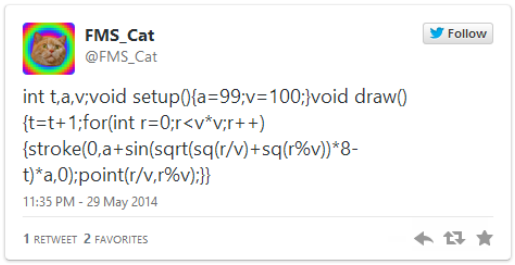


図 7　@FMS_Cat による図 6（1）を表示するコード（140 文字）

Fig. 7　The code drawing Fig.6(1) tweeted by @FMS_Cat (140chars).

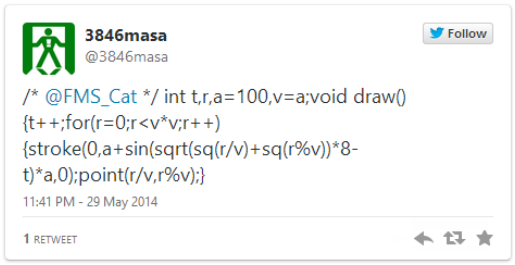


図 8　@3846masa による図 6（1）を表示するコード（117 文字）

Fig. 8　The code drawing Fig.6(1) tweeted by @3846masa (117chars).

て改良が加えられ連鎖的に発展していく点が，特徴のひとつである．

2.2　動的に変化するグラフィック

Twitter でつぶやくことによる利点として，そのままコピーするだけでプログラムが動く手軽さが挙げられる．実例として，@FMS_Cat による動的に変化するグラフィックを紹介する．

図 6（1）は，@FMS_Cat が初めに投稿したツイート（**図 7**）の実行結果である．Processing のグラフィック描画関数の豊富さのおかげで，図 6（1）のようなプログラムを 140 文字以内に収めることに成功している．

その後，そのコードを短縮しただけのコードが@3846masa によって投稿される（**図 8**）．続いて，@FMS_Cat によって図 6（2）が投稿されるが（**図 9**），@3846masa によってさらにコードが縮められて投稿されている（**図 10**）．

図 6 のプログラムを 140 文字以内に収めることの方が，

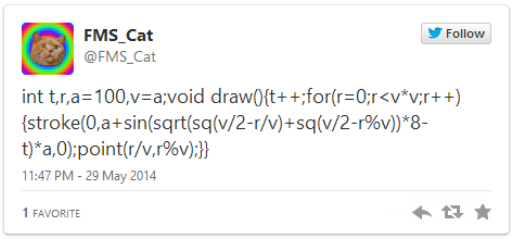


図 9　@FMS_Cat による図 6（2）を表示するコード（125 文字）

Fig. 9　The code drawing Fig.6(2) tweeted by @FMS_Cat (125chars).

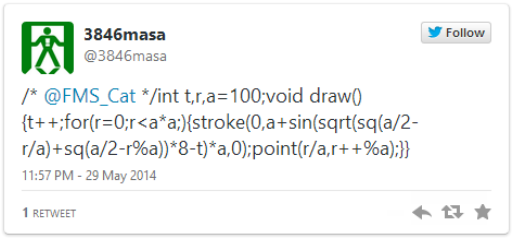


図 10　@3846masa による図 6（2）を表示するコード（120 文字）

Fig. 10　The code drawing Fig.6(2) tweeted by @3846masa (120chars).

投稿後のコードを縮めるより大変であるにも関わらず，この 140 文字 Processing ではより文字数を縮めた投稿が優位に立てる．簡単に実行できることによって，第三者が優位に立つことが容易であることも面白さの一つといえる．

3.　考察

140 字 Processing で用いられる手法には，変数を可能であれば宣言しない，変数名を 1 文字にする，といったコードゴルフやショートコーディングでも用いられる基本的なものから，Processing 特有の命令を用いるプログラムも見られた．

コードゴルフとは，『可能な限り少ないストローク数 (打鍵数，バイト数) で要求仕様を満たすプログラムを作成する遊び』[5] のことである．浜地によれば，コードを短くする遊びは Perl コミュニティを発祥としており，これを Perl 以外の Ruby，PHP，Python でも遊べるようにした Web サイトによって，多数の日本人が参加したという．また，ACM/ICPC の練習用サーバで，自発的に C のコード短縮化を競う運動が北京大学を中心に起こったのが，ショートコーディングの起源とされている [6]．

Processing はこれらの言語と異なり，視覚的な表現をより簡易に行うことに優れている．また，マウスの押下状態を示す mousePressed 変数など，インタラクティブな要素がシンプルに処理できる言語体系を持っている．本稿での課題や事例は，例えば C 言語のコードゴルフでは題材にはならなかったものであろう．

140 字 Processing では，Twitter に投稿するため 141 字以上のプログラムが認められないという制約も興味深い．

つまり，相対的に誰かよりも短いことではなく，ツイートできるか否かという条件が大前提となる．そして，ツイートされたプログラムは多くの人に知れ渡り，後続者にそのソースコードを参考にして追随することを可能にする．最終的にどこまで短くできるかという競技性がある一方で，まずその課題を 140 字以内で実現し，提供する先駆者の存在が非常に重要である．些細なミスやちょっとした工夫などでより短いプログラムを実現されることは，確かに悔しさがあるだろうが，思いついたらすぐ参加でき，すぐに暫定 1 位になれるというのはこの 140 字 Processing ならではの面白さである．

コードゴルフやショートコーディングでは，競技サイトやイベントなど競技場といえる場所が用意されていることが一般的である．しかし，140 字 Processing では日常的な Twitter というコミュニケーションの場が「競技場」となっている．また，誰かが開催を宣言したわけでもなく参加者を募っているわけでもない．Processing を知っている者ならば誰でも参加することができ，知らず知らずのうちにフォロワーを巻き込む，といった広がりも生まれるだろう．こういったことから，この 140 字 Processing は「緩やかな競技」であるといえると考えられる．

筆者らとしては，この論文を通して Twitter 上でのプログラミングの新たな遊びの魅力が少しでも伝われば幸いである．

参考文献

[1] 先端メディアサイエンス 未来を予測するために未来を創る, http://www.fms-meiji.jp/
[2] GitHub, https://github.com/
[3] SECCON, http://www.seccon.jp/
[4] ABPro, http://abpro.jp/
[5] 浜地慎一郎, Code Golf, http://shinh.skr.jp/dat_dir/golf_prosym.pdf
[6] Ozy, Short Coding ～職人達の技法～, 毎日コミュニケーションズ, 2007.