目白大学　総合科学研究
13号　2017年3月　179－185

ものづくりを支援するツールとしてのプログラミング言語の比較検討

Comparison of Programing Language as a Tool to Support Making Things

秋本　結衣
（Yui AKIMOTO）

Abstract:

Programming has been attracting attention in education recently. Programming is also useful as a tool to experience the pleasure of making things that is to create something, thinking and sense of accomplishment, because we can freely realize our thoughts on a computer. In this study, we compared the features of the visual programming language and those of the text-based programming language paying special attention to their functions of a tool to raise the intellectual curiosity to make things in the education of programming in higher education. Visual Programming language is very useful as an introduction of the programming learning because we can create a simple game and an animation without the knowledge of programming skill. However, the text-based programming language can be better than the visual programming language in the degree of freedom, feasibility and development to realize our hopes and dreams freely.

キーワード：プログラミング、ものづくり、教育、ビジュアルプログラミング
Keywords：Programing, Fabrication, Education, Making, Visual Programming Language

1．はじめに

近年、プログラミングが教育現場で注目されている。文部科学省では初等中等教育段階においてプログラミング教育を推進しており、2020年には初等教育でのプログラミング教育の必修化を検討すると発表した[1]。高等教育段階においても、一般科目としてプログラミングを導入する事例が増えており、これに伴いプログラミング教育に対する研究や教材開発が急速に進められている。中でもビジュアルプログラミング言語と呼ばれる、コードを記述しなくてもプログラミングをすることが可能な言語が特に注目されている。これは、コードの記述が必要なく、システム上にあらかじめ用意されているブロック型の部品を組み合わせることで容易にプログラミング体験が可能なため、プログラミング入門段階において有用である。こうしたプログラミング教育の導入の目的は、学校段階ごとに多様であるが、その多くが「プログラミング的思考を身に付ける」ことを挙げている。文部科学省で開催されている「小学校段階における論理的思考力や創造性、問題解決能力等の育成とプログラミング教育に関する有識者会議」によれば、「プログラミング的思考」とは、『自分が意図する一連の活動を実現するために、どのような動きの組合せが必要であり、

あきもとゆい：目白大学情報教育研究室

一つ一つの動きに対応した記号を、どのように組み合わせたらいいのか、記号の組合せをどのように改善していけば、より意図した活動に近づくのか、といったことを論理的に考えていく力』[2] であり、プログラミング教育を通してこのような資質・能力が育成されると考えられている。

一方で、プログラミングは本来コンピュータ上で自由に自分の思いを実現させることが可能であることから、ものづくりの面白さを体感できるツールとしても有用であると考えられる。ものづくりをすることは、創造力や集中力、忍耐力を養うことが期待されるとして初等教育においては「理科」や「図画工作」、中等教育においては「技術・家庭」の中にものづくり教育が取り入れられている。プログラミング教育を通して、コンピュータ上で自由に自分の思いを実現させることの面白さを体感することは、ものづくり教育としての側面も兼ね備えていると言える。プログラミングにおけるものづくりの面白さとは、その自由度の高さ、実現の容易さ、発展性などが挙げられる。コンピュータ１台でものづくりが可能なことから、現実世界におけるものづくりよりも身体的・物理的制限が少ないのも利点である。そこで本論文では、高等教育におけるプログラミング教育の意義の一つとしてものづくりの面白さを体感するツールという視点から、ビジュアルプログラミング言語とテキストベースでコードを記述する従来のプログラミング言語の特徴を比較検討する。

２．先行研究

ここで、プログラミングをツールとしたものづくりについて述べる前に、各教育段階におけるものづくり教育の意義や効果について整理する。

２.１　ものづくり教育の意義

初等中等教育におけるものづくり教育について、「児童生徒のものづくりの教育及び中学校技術科教育に対する意識」を調査した土井ら（2000）は「ものづくりを多く経験したと意識する中学生・高校生は、ものづくりに対する技能・知識への意欲、働いている人たちへの共感の意識、問題解決への意欲、高等学校への技術科教育の期待、熱中してものづくりをする意欲、共同して製作する意欲などが明らかに高く、さらにものづくりが好きである」ことを明らかにしている。また、「ものづくりの豊富な経験量の意識が労働観の育成に好影響を及ぼしている」との調査結果も明らかにしており、ものづくりを経験することで児童生徒らに好影響が及ぼされていることが示されている[3]。

２.２　高等教育におけるものづくり教育

より専門性の高い高等教育においては、「ものづくり」の言葉の持つ意味が初等中等教育とは若干異なっている。職業教育や技術者育成、研究対象としての意味合いが強いからである。しかし中西（2012）は高等教育での「プロ養成」と「研究対象」以外のものづくり教育についての有用性を以下のように述べている[4]。

・ものづくりをすることは、非常に考えたり工夫したりすることが求められ、「楽しさ」が意欲を高めることにつながっていると考えられる
・ものづくりを楽しむことで得られる人間の意欲は、熱意を高め、気づかない間に通常以上の努力をさせ、そのことから「工夫する力」や「創造力」「集中力」「実践力」「諦めない心」「豊かな感性」を育むことが可能である
・それは、初等教育だけでなく、高等教育機関においても重要である

この重要性について、次の二つが考えられる。

一つは、高等教育での学習は、「自ら疑問を持ち、それを追求すること」を求められる。初等中等教育で受けてきた受動的な学習とは異なり、自分で問題を発見し、それを探求する力を養う必要がある。単なる「作品作り」から、より身近な「もの」で、原理を追求できる「ものづくり」であることが、高等教育でのものづくり教育に求められる。たとえば、スマートフォンやパソコン、デジタルカメラなどの電子機器は昨今の大学生にとって非常に身近なものになっているが、それらが「なぜ動くのか」という

原理を理解しながら使用しているのはごく一部の学生に留まるであろう。これらのものをユーザーとして使用するに留まらず、実際に自らの手で作るとなれば、まず情報機器が「何で作られていて」「どのように動いているか」を理解することから始まり、実際に自分の手で組み立てることで内部の構造を知り、思い通りにいかない部分を探すことで「ある問題に対する原因の切り分け作業」を身に付けることが出来る。原理を知ることで、応用できる力を養うことが出来る。たとえば、「パソコンの画面が映らない」という問題が発生した時に、単に「パソコンが故障した」と判断するのか、「ハードウェアの故障である」「ソフトウェアの故障である」と切り分けて判断するかでその後の対応は変化するであろう。原理を知るということは、表面上の形が変わっても原理を応用して対応できる力を養うということにもつながると考えられる。

二つ目は、前述のように、ものづくりの経験量が労働観の育成に好影響を及ぼしているということは、高等教育段階においても同様の効果が期待できる。社会へ出る前の最終過程である高等教育では、初等中等段階の漠然とした労働観とは異なり、就職活動を目前に控えているため、より実感を伴って社会に出て働くことを意識しながら学習させることが出来るだろう。一方的に与えられる講義形式と異なり、ものづくりは、自分の目的に向けて自らの手で対象物を変化させていく作業である。それにはゴールを見据えた設計を行い、目標達成までの道のりを試行錯誤しながら進めるという一連のプロセスがある。自分で考え自分で行動することで主体的に物事に取り組む能力が育まれ、その後社会に出てから必要とされる主体的な姿勢が身に付くことが期待される。

2.3　ビジュアルプログラミング言語を活用したプログラミング学習

次に、ものづくりとプログラミングの関係性について述べる。

Scratchとは、MIT Media LabのMitchel Resnickが開発したビジュアルプログラミング言語である。システム上に用意された「イベント」「制御」「演算」などのブロック型のスクリプトをパズルのように組み合わせることでプログラミングが可能で、画面上に存在するキャラクターを動かすことが出来る。

近年開発されているビジュアルプログラミング言語を代表する存在であり、初等教育でのプログラミング教育を始め、高等教育におけるプログラミング学習の導入としても活用される事

図1　Scratch (https://scratch.mit.edu/)

例が増えている。

Scratchでのプログラミング授業に対して、子どもたちはおおむね好印象を抱いている。森（2011）の研究では「Scratchを使ったプログラミングへの興味について，楽しさの観点からアンケートを行った（n＝36）．５段階［５：楽しかった～１：つまらなかった］で評価をしてもらったところ，平均4.78（.SP：O. 76）の高い評価を得た」[5] との結果が明らかにされている。またScratchと同様に、NTTコミュニケーション科学基礎研究所が開発したビジュアルプログラミング言語「VISCUIT」を用いた授業を行っている小山（2015）は、子どもがプログラミングの授業に抱く楽しさに着目し、授業を受けた子どものうち約90％がプログラミングの授業を肯定的に受け止めていることを明らかにした。小山の調査では、子どもたちはプログラミング授業を「楽しい、おもしろい」と感じていることの他、「もっとこういう部品を使いたい」「もっとこんな作品も作ってみたい」[6] のような部品に対する不満や希望を抱いていることも分かった。これは、ビスケットでプログラミングを行ったことでプログラミングに対しての興味・関心が湧き、さらに作品を発展させたいという欲求の表れであると考えられる。こうした事例から、ビジュアルプログラミング言語はプログラミング導入段階で成功を収めていると言える。

しかしながら、高等教育における「ものづくりの面白さ」は、小学生や中学生と同じように単に部品と部品を組み合わせて作品を完成させて達成感を味わうだけでは不十分である。ものづくりを通して「考えたり」「工夫したり」することで面白さを感じさせることが重要であり、さらに「問題提起する」ことで「深める」ことが可能な題材である必要がある。特にプログラミングをツールとしてのものづくりは、今後社会に出てIT機器に触れる機会の増える学生たちが、実際に社会で動いているソフトウェアの仕組みを学ぶことにも繋がるため、最終的には実社会で運用されているプログラムの一端に触れられるようなものづくりが望ましい。

次の章で、ものづくりを体感するツールとして、ビジュアルプログラミング言語とテキストベースでコードを記述するプログラミング言語のどちらがよりものづくりの面白さを体感できるか明らかにする。

３．ビジュアルプログラミング言語とテキストベースのプログラミング言語の比較

兼宗ら（2009）は「コンピュータは粘土のように何でも作れる」「コンピュータは粘土のように楽しい」ことを子どもたちに伝えるための研究をしており、子どもたちにプログラミングを体験させることで「コンピュータは粘土である」ということを体験させようとしている[7]。兼宗らは、ビジュアルプログラミング言語「LOGO」を用いて子どもたちにプログラミングの楽しさを伝えようと試みているが、テキストベースでコードを記述する従来のプログラミング言語に比べると、ビジュアルプログラミング言語は粘土というよりは、積み木遊びに近いものであると言えるだろう。あらかじめ用意された形の部品を使い、部品と部品を組み合わせることで一つのものを作成するという工程が非常に似ており、決まった形の部品が用意されているからこそ、コンピュータ上で何でも作れると実感させるには至らないと考えられる。プロ

表１　主なビジュアルプログラミング言語

言語名	開発元
Scratch	MIT Media Labo
VISCUIT	NTTコミュニケーション科学基礎研究所
MOONBlock	秋葉原リサーチセンター
プログラミン	文部科学省
Google Blockly	Google

グラミングをツールとして「コンピュータ上で何でも作ることが出来る」ということを感じてもらうためには、ビジュアルプログラミング言語にはいくつかの壁があるように考えられる。

本章では、プログラミングの自由度・実現性・発展性に着目し、3つの壁について検討する。

3.1 自由度の壁

プログラミングの面白さの一つに、「自分の思いが自由に実現できること」が挙げられる。粘土のように、自分の手でコンピュータを変幻自在に操ることで自分のアイディアを実現することが可能である。実際に、コンピュータ上で動作するアプリケーションは全てプログラミングで作られており、今や身近な存在となった携帯電話やスマートフォン、家電製品なども全てプログラミングによって動いている。こうした、自分の思いを実現するプログラムを作成するとき、コーディングによるプログラミングではまず、アイディアを形にするのに適した言語を選択する。あるいは部分によって言語を変え、多様な言語を組み合わせることでそれが実現可能となる。例えば、webアプリケーションと呼ばれるブラウザ上で動作するプログラムは、PHPやASPなどのプログラムの部分とHTMLやCSSといった視覚的な表現の部分、そしてJavaScriptなどの操作上の仕掛け部分から成っているが、そのほとんどがテキストベースで記述されて作成されている。HTMLやCSSの部分がエディタを使用して視覚的に作成される場合を除けば、コーディングなくしてこのようなアプリケーションは作成できないのが現状である。テキストベースでのプログラミングを習得することは、ビジュアルプログラミングには再現できないプログラムの作成を可能にする。そしてテキストベースでのプログラミングは、一度プログラミング記述のルールや概念を学習してしまえばいかようにも汎用が可能である。例えば、PHPという言語はサーバ上で動作し、webページを動的に生成することを得意とする言語であるが、JavaScriptというウェブブラウザ上で動作する言語と基本的な記述のルールはほとんど変わらない。表現や変数の宣言方法など多少の違いはあるが基本的なプログラミングの考え方やルールは同じであり、webアプリケーションもデスクトップアプリケーションも使用する言語を変えることで開発が可能である。スマートフォンやパソコンで当たり前のように使っているアプリケーションがどのように作られどのように動いているのかを知ることで、ユーザーとして完結していた意識から、開発者としての考え方を意識させることができ、独自のアイディアを自由に形にできることの面白さや達成感を感じさせることが可能である。さらに、自分が作成したアプリケーションを自分で活用したり、周囲の人に活用してもらうことで、自分が作ったものを使ってもらえる喜びを味わうことができ、次のものづくりへの意欲につながることが期待される。

3.2 実現性の壁

ものづくりにおいて、実現の容易さは重要である。あまりにも複雑で難解なものづくりは、その楽しさを感じる前の段階で苦痛を感じさせてしまい、挫折へと繋がる可能性があるからである。ビジュアルプログラミング言語の多くは子ども向けに開発されていることもあり、非常に簡単にプログラミングが可能であることが強みの一つである。しかしながら、自分のアイディアを実現することが容易であるかといえば、一概にそうは言えない。デジタルネイティブと呼ばれる、幼い頃からデジタル機器に慣れ親しんできている人々にとって、コンピュータやゲーム、携帯電話などはごく当たり前に身近に存在するもので、プログラミングで出来ることがソフトウェアの開発やゲームの開発、webページの作成などと結びつくであろうことが考えられる。初等中等教育においては、ビジュアルプログラミング言語を使った授業で簡単なゲームやアニメーションを作成したり、ゲーム感覚で遊べる問題を解くことを題材にしている。ビジュアルプログラミング言語のほとんどは、こうした題材に非常に適しているからである。しかしながら、高等教育におけるものづくりは、初等中等教育と同じように「楽しいおもちゃ」を

作るだけでなく、それが実社会と結びつくことを意識できるようなものである必要がある。単にものづくりが「楽しい」だけでなく、ものづくりを通して学習意欲を高めさせ、さらに深めることが出来る題材が求められる。このような、実社会と結びつくようなものづくりの題材としては、Webアプリケーションの作成であったり、スマートフォンで実装されているようなネイティブアプリケーションの開発が適していると言えるだろう。

一方で、コーディングによるプログラミングの難しさが挫折へと繋がることが懸念されるため、高等教育においてもビジュアルプログラミング言語を用いてプログラミング教育を行う事例が多数存在する。プログラミング習得の難しさには独特の文法や構文エラーが挙げられるが、高等教育においては、学習を易しい方へと導くのではなく、難しいからこそ試行錯誤することの面白さを伝えるべきである。プログラミング学習の導入として、体験的にビジュアルプログラミング言語を用いたプログラミングを行うことで学生の興味を引くには有効であると考えられるが、より深くまで「プログラミングでのものづくり」を探求させるには、自らコードを書き、実行し、エラーが発生すれば原因を特定し、修正する、という一連の試行錯誤的な作業は、むしろ必要なものであると考えられる。その試行錯誤が完成した時の達成感につながり、その難しさが面白さへとつながるのではないだろうか。

3.3　発展性の壁

テキストベースでコードを記述することの強みの一つに、その発展性が挙げられる。コーディングによって作成されたプログラムは、そのコードの一部を複製することでいくらでも作品を発展させることが可能である。コーディングによるプログラミングは、同じ動作をするプログラムでも記述の仕方に正解がなく、いかようにも記述することが可能である。自分の思い通りの動作をさせるために試行錯誤して記述されたコードは、自分で作成した「部品」として蓄積していくことが可能で、自分で作成した「部品」と「部品」を組み合わせることで新たにものづくりを楽しむことが出来る。同じ「部品」でも、自分で作った部品であれば、部品自体も変形させることが容易である点がビジュアルプログラミング言語より汎用性が高いと言えるだろう。

さらに、データベースと連携させることで、プログラミングによるものづくりはより豊かな発想が可能となる。たとえば、PHPとACCESSデータベースを連携することで、独自のブログシステムを開発することが可能である。単純なブログシステムの作成でも、既存のシステムに頼らず独自に開発することで自由なレイアウト表示が可能なことや、必要な機能だけを実装することで動作を軽快にし、非常にシンプルな設計にすることも可能である。記事の投稿方法についても、独自のフォームを作成することで使いやすいようにカスタマイズすることが可能である。こうしたシステムを開発するにあたっては、設計から自分で考える必要があり、アイディアを実現させるために必要な部品を洗い出すことや、どのように部品をつなぎ合わせていくかを考えさせることで、物事を体系的に考える力にも繋がることが期待される。

ビジュアルプログラミング言語の中には、拡張機能としてビジュアルで作成したプログラムをテキストベースのコードに変換できる機能が備わっているものがある。Google社が開発したプログラミング言語BlocklyやMOONBlockなどがこの機能を備えている。これは、ビジュアルプログラミング言語でのプログラミングをテキストベースでコードを記述するプログラミングに結び付けるための機能であり、ビジュアルプログラミング言語でのプログラミングを導入として、将来的にはコーディングによるプログラミングに発展させていくことを示唆しているのではないだろうか。

4．おわりに

4.1　まとめ

高等教育におけるものづくりを支援するツールとして、ビジュアルプログラミング言語とテキストベースでコードを記述する従来のプログ

ラミング言語の特徴を比較した。結論を以下に示す。

・ビジュアルプログラミング言語は、プログラミング学習の導入部分としては非常に有用であり、初等中等教育段階においてはこれを用いることで簡単なゲームやアニメーションを作成するなどしてものづくり体験をすることが出来る。
・高等教育段階における「自分の思いを自由に実現する」ツールとしては、自由度や実現性、発展性の部分でテキストベースで記述する従来のプログラミング言語の方がよりものづくりの面白さや達成感、「コンピュータ上で何でも作れること」を感じさせることが可能である。

コンピュータ上でのものづくりは物理的制限を受けにくいことから非常に自由度の高いものづくりが可能で、粘土のようになんでも作ることが可能である。プログラミングを身に付けるということは、コンピュータを自由自在に楽しむことが出来るということである。プログラミングを学習することを、身構えて難しく捉えるのではなく「ものづくりを楽しむためのツール」として面白く感じてくれる学生が増えれば良いというのが筆者の希望である。

今後は、実際にプログラミング教育を受けている学生に対し、ものづくりを意識させたプログラミングとしてビジュアルプログラミング言語とテキストベースのプログラミング言語の両方を体験してもらい、今回示した結果の妥当性を検証していく。また、コーディングによるプログラミングの利点を生かすまでのレベルに学生をどのように教育するか、という議論については今後の課題としたい。

【引用文献】

［1］高浜行人、小学校でのプログラミング教育必修化を検討　文科省、朝日新聞DIGITAL、http://www.asahi.com/articles/ASJ4M5D4GJ4MUTIL044.html、2016年4月20日発行、2016年9月16日閲覧
［2］文部科学省、小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）、http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm、2016年9月16日閲覧
［3］土井康作、奥野信一、横尾恒隆、坂口謙一、田中喜美、近藤義美、木村真、角和博、森山潤、長谷川雅康、児童生徒のものづくりの教育及び中学校技術課教育に対する意識―小学校3年生～高等学校3年生を対象とした10都県の意識調査―、産業教育学研究、30（1）、57-63、（2000）
［4］中西眞弓、高等教育機関におけるものづくり教育の現状と課題、神戸山手短期大学紀要、55、39–48、（2012）
［5］森秀樹、杉澤学、張海、前迫孝憲、Scratchを用いた小学校プログラミング授業の実践：小学生を対象としたプログラミング教育の再考、日本教育工学会論文誌、34（4）、387–394、（2011）
［6］小山万作、ビスケットを使ったプログラミング指導：児童はプログラミングの授業のどんなところを楽しいと思うのか（情報教育の新しい流れ，課題研究，教育情報と人材育成～未来を育む子供たちのために～）、日本教育情報学会年会論文集、31、166–169、（2015）
［7］兼宗進、阿部和広、原田康徳、プログラミングが好きになる言語環境、情報処理、50（10）、986–995、（2009）