

IchigoJam用ビジュアルブロックプログラミング環境の 開発とプログラミング体験教室の実践

鴻池 泰元^{†1,a)} 中西 通雄^{†2,b)}

概要: 2020年からの小学校でのプログラミング教育必修化に向けて、様々なプログラミング初学者向けの学習環境が開発されてきている。本研究では、こども向けプログラミング専用パソコンであるIchigoJam用のビジュアルブロックプログラミング環境を開発した。これを少しずつ改良を加えながら、2017年5月より、プログラミング体験教室を4回実践してきた。その結果として、開発したビジュアルブロックプログラミング環境は、IchigoJamを使ったプログラミング教育ツールの一つとして十分なり得ると考えている。

キーワード: プログラミング教育, ビジュアルプログラミング, Blockly, IchigoJam

Development of visual block programming environment for IchigoJam and its use in introductory programming exercise workshops

TAIGEN KONOIKE^{†1,a)} MICHIO NAKANISHI^{†2,b)}

Abstract: Many programming environments for novices are being developed, since programming education becomes compulsory at elementary school in 2020. In this research, we have developed a visual block programming environment for IchigoJam which is a tiny computer and is programmable in BASIC. We had four programming workshops for elementary school students in 2017 and have been improving our software and curriculum little by little. As a result, I think our visual block programming environment is useful for beginners.

Keywords: Programming education, Visual programming, Blockly, IchigoJam

1. はじめに

2020年からの小学校でのプログラミング教育必修化に向けて、プログラミング初学者（以下、初学者）向けに様々な学習環境が開発されてきている。なかでもよく用いられているのは、Scratchに代表されるビジュアルブロックプログラミング環境である。ビジュアルブロックプログラミング環境を用いることで、「キーボード操作がおぼつかない

子供にとっては、テキストベースの言語は敷居が高い」[1]という問題を解決することができる。

また、初学者にとっては、実行結果がパソコンの画面に表示されるだけよりも、センサを使ってモータを回したりLEDを光らせたりするような、実際に目に見えるものをプログラムで制御する学習環境の方が楽しく学べると考えている。実際に目に見えるものを扱う既存のプログラミング学習環境として、なのぼ〜どAG, KOOV, micro:bit, アーテックブロックなどがあるが、本研究ではIchigoJamに着目した。

IchigoJamは、初学者向けのマイコンボードである。本体が1つ1,500円と安価でありながら、センサやアクチュエータを簡単に制御できる。ここ2年以内に限定して見て

^{†1} 現在, 大阪工業大学 大学院情報科学研究科
Presently with Osaka Institute of Technology Graduate School of Information Science

^{†2} 現在, 大阪工業大学 情報科学部
Presently with Osaka Institute of Technology Information Science Division

a) m1m17a05@st.oit.ac.jp

b) michio.nakanishi@oit.ac.jp

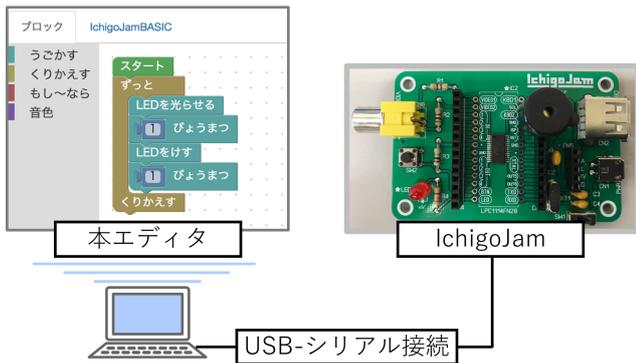


図 1 本エディタと IchigoJam の接続

Fig. 1 Programming editor and IchigoJam



図 2 エディタ画面

Fig. 2 Editor Screen

も、インターネット上での報道 [2][3] や、研究発表が複数あり [4][5][6]、初学者向け教材として注目されている。

しかし、IchigoJam にはビジュアルブロックプログラミング環境は存在しないため、BASIC のプログラムコードを入力する必要がある、初学者は慣れないタイピングを多く行わなければならない。本研究では、IchigoJam 用のビジュアルブロックエディタ（以下本エディタ）を開発し、2017年5月より、小学3年生～6年生を対象とした簡単なプログラミング体験教室を4回実施した。

小学生にテキスト入力によるプログラミングを実践している事例 [7] もあるが、小学生には英語表現のプログラムコードを理解することは難しいと考えられる。日本語で表現されたビジュアルブロックプログラミング環境を用いることで、小学生に対応することが狙いである。これまでの著者らの研究グループでの実践からも、日本語表現のビジュアルブロックプログラミング環境で、センサやアクチュエータを制御できる学習環境が、小学生のプログラミング教育用教材に適していると感じている [8]。

2. IchigoJam 用ビジュアルブロックエディタ

2.1 実現方式

本エディタは、ビジュアルブロックエディタを作成するための JavaScript 用ライブラリである Blockly [9] を用いている。また、Google Chrome ブラウザ（以下 Chrome ブラウザ）上で動作する Chrome アプリとして実装している。

本エディタをダウンロードしたパソコンを、USB シリアルモジュール（図 3）を用いて IchigoJam と接続する（図 1）。

プログラミングを行うために使用するブロックには、それぞれに対応した IchigoJam 上での命令コマンドの文字列が設定されている。「送る」ボタン（後述 2.3 節）がクリックされることで、組み合わせさせた文字列が 1 文字ずつ IchigoJam へ送信される。IchigoJam からは、高速でキーがタイプされているように見え、プログラムの送信が完了する。



図 3 USB シリアルモジュール

Fig. 3 USB serial module

2.2 動作環境

Chrome アプリとして実装しているため、OS には依存しない。Chrome ブラウザのみインストールを必要とする。Chrome ブラウザのインストールができない環境であれば、ファイルをダウンロードするだけで Chrome ブラウザが使用できるポータブル版 [10] でも使用可能である。なお、USB シリアルモジュールを動作させるためには、ドライバのインストールが必要である。

2.3 使用方法

本エディタを起動すると図 2 の画面が表示される。各ブロックは命令の特性（アクチュエータを動かす、反復、分岐、音色）ごとに 4 つにカテゴリ分けしている（図 2 の①）。各カテゴリ名をクリックすると、そのカテゴリに属するブロックの一覧が表示される。

ブロックはクリックまたはドラッグ&ドロップすることで、プログラミングを行うフィールド上（図 2 の②、以下フィールド）に置かれる。フィールド上でブロックを繋ぎ合わせることでプログラムを作ることができ、プログラムが完成したら、「送る」ボタン（図 2 の③）をクリックすることで IchigoJam へ BASIC コードが送信される。

なお、送信される BASIC コードは、「IchigoJamBASIC」タブ（図 2 の④）をクリックすることで画面を切り替えて確認できる（図 4）。BASIC コードを表示できるようにすることで、コンピュータが実際に受け取っている命令はどのようなものか確認することができる。なお、図 4 を見てわかるとおり、BASIC で永久ループを実現するためにこのようなコードを生成している。本エディタでは、反復処理を行うブロックは全て FOR 文が出力されるようにして



図 4 BASIC コード出力画面
 Fig. 4 BASIC code output screen

いる (後述 3.2 節).

IchigoJam へプログラムを送信した後は、「実行」ボタン (図 2 の⑤) や「停止」ボタン (図 2 の⑥) をクリックすることで、IchigoJam 上でのプログラム実行、停止を制御できる。「実行」ボタンをクリックしたときには、IchigoJam 上での実行コマンドである「RUN」の文字列を送信する。また、「停止」ボタンではプログラム停止コマンドである ESC キーの文字コードを送信する。

また、作成したプログラムは「保存」ボタン (図 2 の⑦) をクリックすることで XML ファイルとして保存できる。保存した XML ファイルは「開く」ボタン (図 2 の⑧) をクリックしてファイルを選択することで保存した時のブロックを再現できる。

3. 本エディタの特徴

本研究では、プログラミング初学者は、簡単な命令のみを用いてプログラミングを体験した方が、より楽しみながら学べると考えている。この考えの元に本エディタは以下のように設計・実現した。

3.1 ひらがな表記

前述したように我々が実施しているプログラミング教室では、対象を小学 3 年生～6 年生に定めているので、小学 3 年生以上で学ぶ漢字のみを使用した。

3.2 変数の概念の排除

BASIC における FOR 命令をそのままブロック化すると (図 5)、変数の概念を教えざるを得なくなる。従って図 5 のようなブロックは見せずに、変数の概念を排除した指定された回数反復するブロックを用意した (図 6)。

2 重ループ、3 重ループなどにも対応しており (図 7 も同様)、実際にループに使用する変数名は I, J, K... と自動で設定される。単純な反復処理では GOTO 命令とラベル



図 5 FOR ブロック
 Fig. 5 FOR block



図 6 変数を無くした
 反復ブロック
 Fig. 6 Loop block
 without loop
 variable

図 7 永久ループブロック
 Fig. 7 Infinite loop
 block

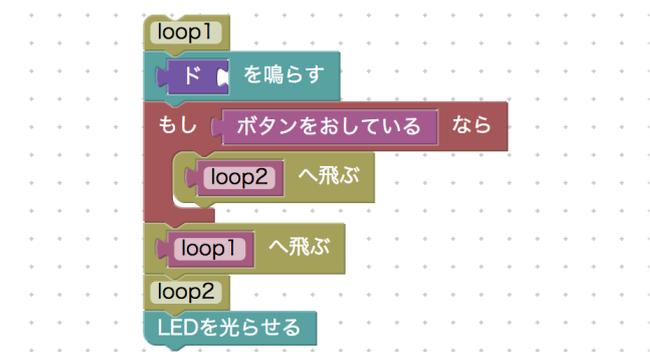


図 8 GOTO を使ったプログラム例
 Fig. 8 Example program using GOTO

を使った方がすっきりと記述できるが、もし今後 GOTO 命令にあたるブロックを実装した場合に、ラベル名が競合する可能性を考慮し、FOR 命令が出力されるようにした。

3.3 反復構造の明確化

BASIC における GOTO 命令をそのままブロック化した場合 (図 8) には、プログラムの制御構造が分かりづらくなる可能性がある。簡単なプログラムを体験させる上で GOTO を使った無限ループはよく用いられるが、より制御構造が分かりやすいブロックを用意した (図 7)。

3.4 条件式の日本語化

センサの値を用いたプログラミングをする場合に、条件式 (図 9) で表すよりも、日本語で表現した方が分かりやすい。本研究で実施したプログラミング教室の内容に合わせて、センサの値を用いた条件式を、日本語化したブロックを用意した。



図 9 条件式のブロック



図 10 条件式の日本語表現

Fig. 9 Conditional block

Fig. 10 Conditional expression in Japanese



図 13 数値ブロック

Fig. 13 Numeric block



図 14 「ボタンをおしている」ブロック

Fig. 14 "Pushing Button" Block

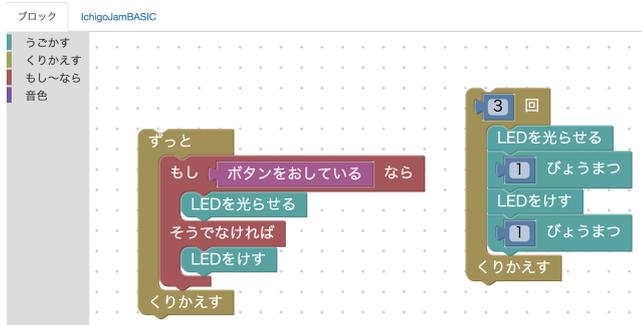


図 11 ブロックの塊の例

Fig. 11 Example of block chunks

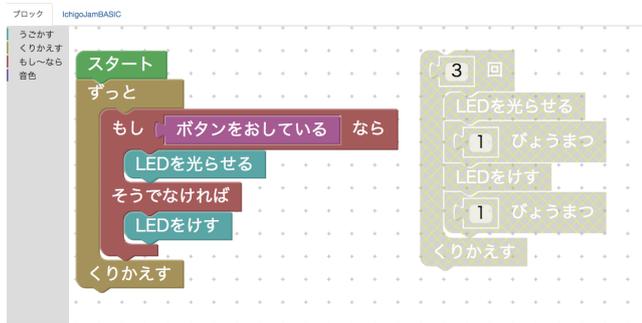


図 12 「スタート」ブロックによる実行対象の明確化

Fig. 12 Clarification of execution target with "start" block

図示したブロックは、光センサを用いるときの条件式である。図 10 のブロックでは「へやが明るい」「へやが暗い」の 2 つの条件を指定できる。センサ値がいくつであれば部屋を明るいとすか、といった値の微調整はできないが、今回のような実践ではそのような微調整は必要なかった（後述 5.2 節）。

3.5 実行対象ブロックの明確化

過去の実践 [11] において、小学生がフィールド上に残したブロックが実行されてしまう様子が見られた（図 11）。これは、Blockly の仕様でフィールド内の座標が高い位置にあるブロックから処理されていた。そこで、「スタート」と書かれたブロックに接続されているブロックのみを実行し、接続されていないブロックは薄い色で表示することで、実行されないことが明示的にわかるようにした（図 12）。

プログラミングをする過程では、複数のコードを書きながらどのプログラムが良いかを試行錯誤することがある。ブロックを繋げ合わせたものを「スタート」ブロックに付け替えながら試すことで、同様の試行錯誤を行うことができる。



図 15 Scratch の数値ブロック
Fig. 15 Scratch's numeric block



図 16 Scratch の条件ブロック
Fig. 16 Scratch's conditional block

表 1 学年ごとの参加者の累計

Table 1 Cumulative number of participants by grade level

	2 年生	3 年生	4 年生	5 年生	6 年生
人数 (人)	1	15	11	5	7

3.6 プログラムエラーの排除

各ブロックは不正な組み合わせができないようになっている。例えば、「数値ブロック」（図 13）と「ボタンをおしているブロック」（図 14）は同じ形であるが、「ボタンをおしているブロック」は図 6 のブロックにはまらないようにしている。

Scratch では、ブロックの不正な組み合わせを防ぐために、そもそも異なる形にしている（図 15）（図 16）。Blockly はそうではなく、当てはめる場所の形は一つであり、はめ込むことができるブロックを指定できる。本来は Scratch のようにするのが理想と考えるが、現状は実装方法が分からないためこのようにしている。

4. プログラミング教室の実践

4.1 対象者・募集方法

体験教室では小学 3 年生以上を対象とした。5 月、8 月、および 10 月の実施では、本学の枚方キャンパス近隣の複数の小学校へ出向いてちらしを配布してもらい、参加者を募った。1 月では、本学梅田キャンパス近隣の 1 つの子供会と、親子向けイベント掲載サイト「いこーよ」[12] から募集した。いずれの回も保護者同伴を必須とした。これまでに本エディタを使用した小学生の学年ごとの人数を表 1 に示す。

4.2 実施体制

実施する内容はスライドショー形式にして著者の鴻池が解説しながら進行したが、スライドショーの内容は印刷して各小学生に配布し、各自が自由に進められるようにした。解説中は、大学生アシスタントが随時巡回しながら質問に答えられるようにした。実施月ごとの、小学生に対するアシスタントの人数を表 2 に示す。アシスタントには、実施する内容や IchigoJam の使い方などを事前に時間を

表 2 実施月ごとの小学生に対するアシスタントの人数

Table 2 Number of students and assistants per month

	5月	8月	10月	1月
小学生(人)	6	10	8	15
アシスタント(人)	2	3	2	10

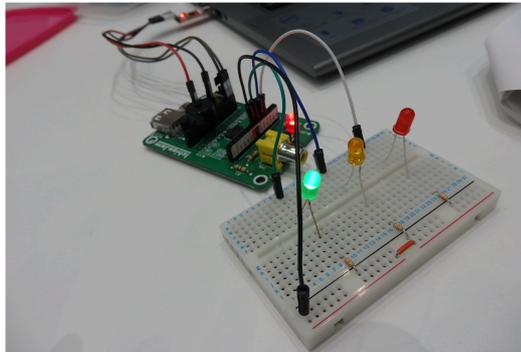


図 17 電子工作の例

Fig. 17 Examples of electronic work

とってレクチャーしておいた。プログラミング教室の実施時間はいずれも2時間30分(休憩5~10分)ほどである。

4.3 実施内容

体験内容は、実施するごとに若干の変更はあるが、基本的には以下の構成である。

- (1) ブロックの取り出し方, プログラム送信・実行方法
- (2) ブロックの繋げ方, WAIT^{*1}の必要性
- (3) ブロックの外し方, LEDを点滅させる(反復処理)
- (4) ブザーを鳴らす
- (5) ボタンを押してブザーを鳴らす(分岐処理)
- (6) 簡単な電子工作

本エディタの使い方を絡めながら、LEDを用いたプログラムの順次・反復処理を体験させ、後にブザーとボタンを使って分岐処理を体験させる。プログラミングの基本をひとつと体験させたところで、ブレッドボード上にジャンプワイヤ、LED、マイクロサーボモータなどを用いて簡単な電子工作を体験させた(図17)。

(2)の「WAITの必要性」とは、指定した秒数のあいだ処理を待つ(図18)という命令を使って、コンピュータが高速で命令を処理していることを体験させる、という内容のことである。初めに、LEDを光らせた後消すプログラムを実行させるが、WAITが無いために一瞬でLEDを消す命令まで実行されてしまう。LEDが光っていた状態を確認できないことによって、コンピュータが高速で命令を処理していることを体験させる。

(6)の電子工作では、内容によっては実施時間内に終わらなかったケースがあったため、回を追うごとに使う機材を変更していった(表3)。

*1 IchigoJam BASICにおける命令の1つ。動作を一時停止させる



図 18 指定秒数処理を待つブロック

Fig. 18 WAIT block

表 3 実施月ごとに使用した機材

Table 3 Parts used per month

	使った機材
5月	5色のLED
8月	5色のLED, 光センサ, マイクロサーボモータ 赤外線センサ
10月	3色のLED, マイクロサーボモータ
1月	3色のLED, マイクロサーボモータ

初回となる5月では、5色のLEDを用いたプログラムを作らせた。8月では、センサとアクチュエータを増やした。10月では、使うLEDの数を減らし、信号機という身近なものを模したプログラムを作るというテーマを与えた。さらに、小学生自身に画用紙に絵を描かせ、マイクロサーボモータと組み合わせて自分の描いた絵を動かすようにさせた。1月では、内容は10月と同じだが、受講者定員を15名に増員して実施した。定員を増やすことで、実際の小学校での授業を想定した知見を得ることを目的とした。

4.4 実施内容の変遷について

5月の実施では、用意した体験内容を先に全て終えてしまった子供がいた。そのため8月では、センサやアクチュエータを増やし内容を拡大したが、内容を増やしすぎたため、殆どの小学生がマイクロサーボモータと赤外線センサを使ったプログラミングを体験できなかった。この反省を踏まえ、10月では3色のLEDとマイクロサーボモータを使った内容に留めた。この改善によって2時間30分という実施時間に収まるようになった。

なかでも、マイクロサーボモータと画用紙を使ったプログラミング体験は非常に好評であった。各々が様々な作品(図19, 図20)を作り、持ち帰る小学生もいた。図19はアニメのキャラクターを動かす作品、図20は隣同士で協力して、魚がワニを食べる様子を表現した作品である。

5. 評価

5.1 アンケート結果について

著者らは2017年度に、プログラミング体験教室(以下体験教室)を4回実施してきた(5月, 8月, 10月, 1月)。本エディタが小学生向けのプログラミング教育ツールとして利用可能であるかどうかを、アンケートや実施中の観察結果を元に評価した。

5月は保護者に対するアンケート(以下保護者アンケート)のみ、10月, 1月は小学生にもアンケート(以下小学



図 19 小学生の作品例 1
Fig. 19 student's work example 1



図 20 小学生の作品例 2
Fig. 20 student's work example 2

表 4 小学生アンケートの回答項目
Table 4 Items of questionnaire to students

質問項目	回答項目
難しかった課題の番号は？	各課題の番号
パソコン操作は難しかったか？	簡単, まあまあ簡単 ちょっと難しい, 難しい
電子工作は難しかったか？	簡単, まあまあ簡単 ちょっと難しい, 難しい
またやりたいか？	やりたい, もっと難しいのが良い もっと簡単なのが良い, もう良い

生アンケート)を実施した。10月の小学生アンケートでは、「難しかった課題の番号は?」「またやりたいですか?」といった旨の項目を用意した。

回答が課題番号だけでは、本エディタに対する操作や電子工作の部分についてどのように感じたかがわからず、実施中の様子から観察することしかできなかった。そこで1月の小学生アンケートでは、これまでの項目に加えて「パソコン操作は難しかったか?」「電子工作は難しかったか?」といった旨のアンケート項目を用意した。それぞれの質問に対する回答項目は表4の通りである。

これまでの保護者アンケートで、体験教室に対する良かった点についての自由記述を一部抜粋する。

- 身の回りの物やプログラミングの仕組みが知れた。
- パソコンはゲームをしたり, YouTube を見たりするだ

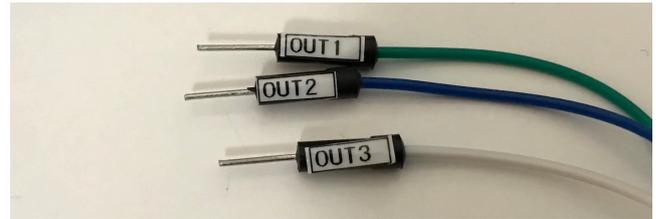


図 21 ラベルを貼ったジャンプワイヤ
Fig. 21 Labeled jump wire

けの物ではないことがわかった

- スマホアプリなどでは画面のキャラクターが動くだけなので、子供がつかめるイメージが限定される。実態のあるものを触ってプログラミングできるのが良い。
- 音が鳴ったり動いたりしたところがよかった
- ソフト上で簡単にブロックで自由にプログラムが作成できる

悪かった点の自由記述は、進行の段取りに対する不満や、マニュアルの図が分かりにくかった、などが挙げられた。マニュアルの図については、ジャンプワイヤの色が実物と写真の色が同じであったり、違ったりしたことが混乱を招いたようであった。ジャンプワイヤの接続先は写真に示していたが、より分かりやすくするために、1月の実施ではジャンプワイヤのピンに接続先を示すラベルを貼った(図21)。これにより、1月のアンケートではマニュアルの図に関する不満はなかった。

10月の小学生アンケートの「難しかった課題の番号は?」に対する回答は、電子工作に関わる課題番号が多かった。1月の小学生アンケートでは、電子工作を少しでも難しく感じた者は15人中4人であった。パソコンの操作に関してはいずれの小学生も「簡単」または「まあまあ簡単」の回答であった。また、いずれの小学生も「またやりたい」「もっと難しいのが良い」と答えた。

5.2 考察

マイクロサーボモータと画用紙による作品づくりでは、小学生がただやらされるだけでなく、自分で何かを作ったという実感を得て、プログラミング体験の印象をより深めている。小学生が手を動かして、絵を描いて自分の作品を作ることで、どのように動かしたいかを考えるようになり、創意工夫を持って取り組んでいたように見えた。

5月、8月の実施では、図10の光センサ用のブロックを使用した。本来、BASICならセンサ値がいくらであれば部屋を明るいとするか、暗いとするか、または明るさの段階はいくつにするか、というように自由に条件式を記述できるが、このブロックは、「部屋が明るい」「部屋が暗い」の2つの条件しか指定できない。しかし、今回の実践を通して、初学者向けのプログラミングではそのような詳細に条件式を編集できる仕組みは必要無いと感じた。

表 5 機材 1 セットあたりの費用
Table 5 Cost of parts per student

機材	価格 (円 (税込))
IchigoJam	1,500
USB シリアルモジュール	1,150
ブレッドボード	280
LED × 3	90
ジャンプワイヤ × 7	245
マイクロサーボモータ	400
電池パック	110
合計	3,775

改善してきた小学生アンケートについて、全ての回を終えてみて、改めて難しかった課題の番号を質問項目にしたのは失敗であったと考える。回答項目には課題の番号ではなく、「LEDの点滅」「ボタンで音を鳴らす」「マイクロサーボモータの配線」といったように小学生が「実際にやったこと」を回答項目に置いた方がより確実な結果が得られたように思う。

これまでの実践で「またやりたいか?」という質問に対しては全員「やりたい」もしくは「もっと難しいのが良い」の回答であったため、プログラミングに対する興味をより強く持たせることができたと感じる。しかし、体験教室の参加者や保護者はそもそも自ら進んで参加しているので、元からプログラミングに対する興味はあったと考えるのが自然である。よって、プログラミングに興味の無い小学生に対しても実施してみることが必要かもしれない。

パソコンの操作に対して難しいと答えた小学生はいなかったため、本エディタは小学生にとっても十分扱えるものであることがわかった。

5.3 小学校に導入する上での課題

5.3.1 使用機材の適切さ

保護者アンケートの自由記述からも、実在する物を使ってプログラミングをすることは好印象であった。保護者から見てもプログラミングに対して良いイメージを得ることができたので、プログラミング未経験の小学校教員にとっても親しみやすい内容であると考えられる。昨今のプログラミング教育必修化に伴う教員不足の懸念に対しても、本研究での取り組みは有用であるかもしれない。

5.3.2 予算面

1セットあたりの機材の費用は、表5に示すとおり4千円弱であった。実際にはこれらの機材とは別にノートパソコンを用意している。小学校ではパソコン教室が設置されているが、システム保護機能が導入されていて、USBシリアルモジュールのドライバをインストールできない可能性がある。費用に関しては、小学生2人につき1セットを用意するとすれば、運用可能と考えられる。



図 22 開発中の独自ブロック追加機能画面
Fig. 22 Defining a new block (under development)

5.3.3 実施体制について

これまでの実践では、アシスタント1人につき小学生を3-4人見るようにしていた。さらに、保護者同伴であったため、保護者がアシスタント代わりになっている場合もあった。実際の授業では、アシスタントの人数を確保する、あるいはアシスタント無しで実施できるようにする必要がある。

6. 今後の予定

6.1 本エディタのインターネット配布

本エディタをインターネット上で配布し、IchigoJamコミュニティなどのプログラミング教室を実施している人(以下運営者)に使ってもらえるようにする予定である。小学生に対して扱いやすいものであるかどうか、プログラミング初学者向けのツールとして有用であるかどうかは、複数の実践事例から明らかになると期待している。

6.2 独自ブロック追加機能

本エディタはこれまで、体験教室で実施する内容に合わせた新しいブロックを、その都度コーディングして用意していた。

例えば、信号機プログラミングでは、「緑のLEDを光らせる」というブロック、ロボットプログラミングでは、「前進する」というブロックを用意したい。

そこで、GUI上でブロックを設計し(図22)、ブロックの定義ファイルを出力できるようにすることで、運営者の負担を減らすことができる。なお、この独自ブロック追加機能はBlocklyのデモツールである「Block Factory[13]」を改変したものである。

6.3 デスクトップアプリケーション化

2.1節でも述べたように、本エディタはGoogle Chromeブラウザ上で動作するChromeアプリとして実装しているが、Chromeアプリは2018年初頭にサポートが終了することが決定している[14]。そこで、現在は、NW.js[15]を用いて、ブラウザを必要としないデスクトップアプリケーションとしての実装に移行している途中である。

7. おわりに

本研究では、初学者向けのマイコンボードである IchigoJam 用のビジュアルブロックエディタを開発した。さらに、4回のプログラミング教室を実践し、本エディタが初学者向けツールとして有用であると感じた。しかし、小学校での実際の授業として運用するには、機材のコスト、アシスタントの人数などの課題が残る結果となった。

いずれの回も、小学生たちは非常に楽しんで取り組んでいたように見えた。図工とプログラミング教育は相性が良いものであると実感した。

なお、本稿はこれまでの実践報告 [11][16] をまとめて、さらに最新の知見を加えたものである。

謝辞 NPO 法人 NEXTDAY の佐々木東先生には、本エディタの初期の版を試用いただき、多くのご意見をいただきました。心から御礼申し上げます。

本研究の一部は、JSPS 科研費 JP16K01141 および 17K01088 の助成を受けた。

参考文献

- [1] 兼宗 進, 阿部和広, 原田康徳: プログラミングが好きなになる言語環境, 情報処理学会論文誌, Vol. 50, No. 10, pp. 986–995 (2009).
- [2] 高橋正和: はんだ付けから始める小学生向け BASIC プログラミング教室, 「IchigoJam」活用して KidsVenture が展開, INTERNET Watch (オンライン), 入手先 <<https://internet.watch.impress.co.jp/docs/news/1031766.html>> (参照 2018-02-20).
- [3] 岩崎宰守: 「タミヤロボットスクール」全国展開, タミヤの工作キットと IchigoJam でプログラミング学習, INTERNET Watch (オンライン), 入手先 <<https://internet.watch.impress.co.jp/docs/news/1059638.html>> (参照 2018-02-20).
- [4] 不破 泰, 時田真美乃, 長谷川理: IchigoJam を用いた小学生プログラミング教育の成果と今後の計画について, 教育システム情報学会第 42 回全国大会, pp. 431–432 (2017).
- [5] 不破 泰, 斎藤史郎, 大手智之, 野瀬裕昭, 鈴木彦文: IchigoJam を用いたこどもプログラミング教室について, 教育システム情報学会第 41 回全国大会, pp. 87–88 (2016).
- [6] 但馬将貴, 香山瑞恵, 小形真平, 橋本昌巳: モデリング学習用モデルコンパイラを用いた IchigoJamBASIC コード生成ツールの開発, 教育システム情報学会第 41 回全国大会, pp. 311–312 (2016).
- [7] 佐藤正範: 小学校の教科に位置付けたテキスト入力型プログラミング言語の導入についての考察, 2017 PC Conference 論文集, pp. 149–150 (2017).
- [8] 野上理沙, 藤林博貴, 中西通雄: Ardublock のタイトルを用いたプログラムによる計測・制御学習教材, 教育システム情報学会 2015 年度学生研究発表会, pp. 67–70 (2015).
- [9] Google Developers: Blockly — Google Developers, Google Developers (online), available from <<https://developers.google.com/blockly/>> (accessed 2018-02-18).
- [10] PortableApps.com: Google Chrome Portable, PortableApps.com (online), available from <<https://portableapps.com/apps/internet/>

- > google_chrome_portable) (accessed 2018-02-18).
- [11] 鴻池泰元: IchigoJam 用ビジュアルブロックエディタの開発と評価, 2017 PC Conference 論文集, pp. 65–66 (2017).
- [12] アクトインディ株式会社: 子供とお出かけ情報「いこーよ」, アクトインディ株式会社 (オンライン), 入手先 <<https://iko-yo.net>> (参照 2018-02-20).
- [13] Google Developers: Block Factory — Blockly — Google Developers, Google Developers (online), available from <<https://developers.google.com/blockly/guides/configure/ios/block-factory>> (accessed 2018-02-18).
- [14] 佐藤由紀子: Google, ブラウザ向け「Chrome アプリ」の提供を 2018 年までに段階的に終了へ (Chromebook 向けは対象外), ITmedia (オンライン), 入手先 <<http://www.itmedia.co.jp/news/articles/1608/21/news012.html>> (参照 2018-02-18).
- [15] NW.js community: NW.js, NW.js community (online), available from <<https://nwjs.io/>> (accessed 2018-02-18).
- [16] 鴻池泰元: IchigoJam 用ビジュアルブロックエディタを用いたプログラミング体験教室の実践, PC カンファレンス北海道 2017 (論文集無し) (2017).