6G-02

みんなのためのビジュアルプログラミング言語「Enrect」の設計

鈴木　遼†　　長　幾朗†

早稲田大学†

Fig. 1　Enrect のエディタ画面

1. はじめに

プログラミングツールはプログラミング教育の現場を支えるインフラであり、「誰もが公平に扱えるように」というアクセシビリティ面での要求は今後ますます大きくなっていくだろう。

MIT の「Scratch 2.0」[1]は、現在最も成功している教育用プログラミングツールの 1 つであるが、そのユーザ・インタフェース（UI）デザインには、使い勝手やバリアフリーの観点から、改善できる箇所が多く残されている。本稿では、著者らが開発する新しいビジュアルプログラミング言語「Enrect」（**Fig.1**）の設計過程で取り組んだ課題と実装の報告を通して、ビジュアルプログラミング言語のアクセシビリティに関する議論を提起する。

2. ローカライズと UI

ブロックベースのビジュアルプログラミング言語をローカライズする際、ブロックのテキストに含まれる引数の順序を並び替えることで、翻訳後のテキストをより自然な表現にできる場合がある。**Fig.2** は Scratch の「abs of -2」及び「letter 1 of Hello」ブロックと、そのブロックの日本語ローカライズ版における表示である。

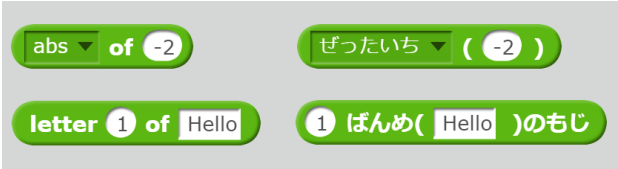

Fig. 2　語順を守ったため日本語訳が不自然な例

このケースでは、英語に準じた「ぜったいち(-2)」や「1 ばんめ(Hello)のもじ」ではなく、引数の使用順を入れ替えて「-2 のぜったいち」や「Hello のもじの 1 ばんめ」という文章にしたほうが、日本語話者には意味が通じやすい。インタフェース設計において語順の変更を許容することで、ローカライズの質を改善できる。

Enrect では、英語版での順番を基準としてブロックの引数に番号を割り振っておくことで、翻訳者が次のように語順を入れ替えて翻訳結果を記述できる形式でブロックを定義している。

```
Random value between {0} {1}
Add {0} to {1}
{0} から {1} のあいだのランダムな数
{1} に {0} をたす
```

また、テキストの置換だけがローカライズにおける考慮事項ではない、翻訳後の文字数の増加によって、テキストの幅がボタンなどの UI コンポーネントの想定を超える場合の表示方法についても、あらかじめテストを行うべきである。

The Design of Enrect, a Visual Programming Language for Everyone

Ryo Suzuki, Waseda University
Ikuro Choh, Waseda University

Copyright ©2018 Information Processing Society of Japan.
All Rights Reserved.

3. 音と UI

効果音や BGM の再生機能、音声ファイルのライブラリを提供する場合、音が出せない環境と、聴覚障害を持つユーザを想定するべきである。

音声ライブラリ上でのファイルの表示については、個々のファイルに適切な名前を付けることで、ユーザに音の種類を想像させることはできる。しかし、例えば「bell」という名前だけでは、呼び鈴か釣鐘かを区別できず、1 回の再生でベルが何回鳴らされるのかもわからない。

Scratch では音声波形を視覚的に表示することで、大まかな音の変化を読み取ることができる。Enrect ではさらに、標準で提供する効果音に絵文字などの画像をタグ付けし、プレビュー時には画像に対して、音のイメージに合わせた、アフィン変換によるアニメーションを付加することで、音の情報を可視化する。例えば、釣鐘の音は低周波で振動するアニメーション、犬が何度も吠える効果音は拡縮を繰り返すアニメーションといったものである。こうした情報はユーザ自身がインポートした音声ファイルにも手作業で付加できるようにしている。

Enrect では、実行中のプログラムが鳴らす音についても可視化する。ツールの設定からサウンドの可視化を有効にすると、実行中のプログラムで再生されている BGM のタイトルとスペクトラムが画面にオーバーレイ表示されるほか、効果音が鳴ったときには対応する画像が画面から飛び出すような視覚効果を描画する。

4. 入力装置と UI

コンピュータの操作形態は多様化している。プログラミング教育の現場では、キーボードもマウスも無いタブレット PC 環境も想定されうる。そうした環境で変数名を入力したり、キー入力をトリガーとするようなプログラムを実行したりするには、OS が提供するソフトウェアキーボードを使う必要がある。しかし、それによってエディタやコンテンツが隠れる問題が発生する。

Enrect では、**Fig.3** のようなエディタと統合した仮想キーボード・ゲームパッドを提供することで、入力装置の有無による差を吸収している。また、Enrect のコーディングでは、変数名の入力が不要なノード・リンクによるフローUI を用いており、1 本指のタッチ操作ですべての開発作業を行えるようになっている。

ところで、「スペースキー」がキーボード上のどのキーを指すのか、パソコンを初めて触るユーザには自明ではない。また、Scratch のキー

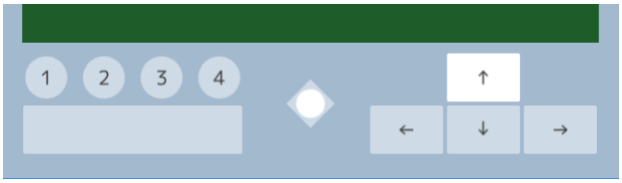

Fig. 3 実行画面の下に表示される仮想キーボード

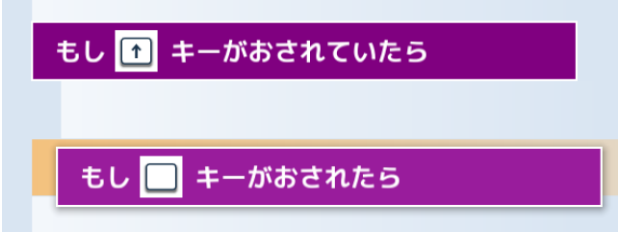

Fig. 4 キーの印字をそのままコード内で再現する

入力に関するブロックでは「a キーが押されたとき」のようにアルファベットがすべて小文字で表記されている問題があり、キーボードに同じ印字のキーが見つからず、非英語圏の学習者が混乱するという声も上がっている[3]。Enrect では **Fig.4** のように、実際のキーボードの印字に合わせたアイコンを使用している。

5. まとめと今後の課題

本稿では、Scratch と Enrect における複数の実装例を紹介しながら、ユーザ及びユーザ環境の多様性に対応するビジュアルプログラミング言語の設計について議論した。プログラミングツールに対するアクセシビリティの要求水準は今後ますます高まり、特に教育用のプログラミングツールの開発者はそれを無視できなくなるだろう。現時点では、ビジュアルプログラミング言語には、Web サイトのアクセシビリティを評価するような体系的なガイドライン[4]が整備されておらず、ツール開発者が各々に模索している状態である。教育現場において、児童生徒に公平なプログラミング体験の機会を提供するためにも、より多くの事例や知見を集約し、効果的な実装を研究していく必要がある。

謝辞

Enrect は JST ACT-I の支援を受けて開発されています。

リファレンス

[1] SCRATCH https://scratch.mit.edu/
[2] Enrect https://enrect.org/
[3] Twitter https://twitter.com/tmk815/status/943055345307459584 (2017)
[4] World Wide Web Consortium. "Web content accessibility guidelines (WCAG) 2.0." (2008).

Copyright ©2018 Information Processing Society of Japan.
All Rights Reserved.