Processing Web IDE を対象としたプログラミング初学者のつまずき防止

Preventing stumbling of novice programmers for Processing Web IDE

三浦 元喜
Motoki MIURA
九州工業大学 基礎科学研究系
Faculty of Basic Sciences, Kyushu Institute of Technology
Email: miuramo@mns.kyutech.ac.jp

あらまし：Javascript 版 Processing を用いた，プログラミング学習環境は，Web ブラウザ上で動作するため初学者向けの演習環境としてすぐれている．しかし，プログラミング初学者は，意図しない文字や無限ループの挿入によって，エラーを含むプログラムを作成してしまったり，Web ブラウザの操作を継続できなくなったりする状況に陥ることがある．我々は，エラーチェックや無限ループをチェックする機能を導入することによって，プログラミング初学者のつまずきを防止し，安心して演習にとりくめるようにした．実際のシステム運用を通じて得られた活動ログによる，学習者の傾向についても報告する．
キーワード：Web 学習環境，無限ループ，エラーチェック

1. はじめに

近年，Moodle や WebClass をはじめとする，CMS が普及している．プログラミング学習においても，Web ブラウザのみで利用できる環境が盛んに開発されている[1,2]．

我々は，C 言語に類似した文法を備えつつ，グラフィックスやアニメーションを簡単に記述することができる Processing 言語に着目し，その Javascript 版実装である Processing.js を利用した Web IDE(統合開発環境)を構築した[3]．そして，大学生を対象とした講義における演習や試験に活用してきた[4]．

しかし，我々が開発してきた Web IDE 上で，学習者が誤って無限ループを含むソースコードを実行してしまうと，ページを表示するブラウザのタブが固まって（フリーズして）しまうことがある．一旦フリーズしてしまうと，ブラウザプロセス全体を終了して起動しなおしたり，ソースコードのリセットを行ったりする必要があった．

そこで我々は，Web ブラウザ上で動作するプログラミング環境の利便性を保ちつつ，初学者がブラウザのフリーズを心配せず，安心して演習に取り組めるようにするため，無限ループの可能性を実行前にチェックする仕組みを導入した．

2. 無限ループのチェック方法

一般に，プログラムの停止性を厳密に検証するのは難しい．我々は，初学者が犯しやすいミスや，陥りやすい点を中心に，正規表現を利用した簡易的なチェック機構を，Javascript の関数として作成した．

なお，Processing では，ループを記述する方法として，while 構文と for 構文が提供されている．今回は，チェックの行いやすさと，利用率を考慮し，for 構文についてのみチェックを行うことにした．

2.1 手順

まず引数で与えられたソースコードに含まれるコメントと，ソースコード中の改行をすべて削除する．次に，‘}’ 文字を，‘}’+‘改行’ におきかえる．その後，for 構文にマッチする文字列について，以下をチェックしていく．

(a) 丸括弧のなかの文字列に，‘,’ (カンマ) が含まれているか？（含まれていたらエラーの可能性）

(b) 丸括弧のなかの文字列に，‘;’ (セミコロン) が 2 つあるか？

（上記 a, b をパスしたら）丸括弧のなかの文字列を，初期化部，継続条件部，更新部に分割する．

(c) 初期化部が，int k = 0 のように，「型」「変数名」「=」「初期値」のみで構成されている？

(d) 継続条件部が，k < 10 のように，「変数名」「比較演算子」「値」を備えている？

(e) 更新部が，変数を増加（または減少）する文になっているか？

(f) (d) における，比較演算子の種類は，(e) における増加(または減少) と整合しているか？

(g) (c) における「変数名」は，(d) や(e) に出現する「変数名」と一致しているか？

(h) (c) における「初期値」は，(d) における「値」や(f) における「増加/減少」の種類と，整合しているか？(なお，int k = 10; k < 10; k++ のように，無限ループにならない場合でも，エラーとして扱う)

3. 文法エラーの提示

Processing.js のコンパイラは，詳細なエラーメッセージを提示せず，かつ厳密なチェックを行わない．そのため，我々はソースコードをサーバに送信し，スタンドアロン版 Processing のコンパイラでチェックする機能を追加した．

4. 実行例

図 1 に，実際に学習者が入力したエラーのプログラムを示す．学習者が実行ボタンを押すと，無限ループチェックによるメッセージが図 2 のように表示される．なお，同一のプログラムをスタンドアロン版 Processing のコンパイラでチェックした結果を図 3 に示す．これは一例であるが，我々が作成した無限ループチェックによるメッセージのほうが具体的であり，学習者は修正しやすいと考えられる．

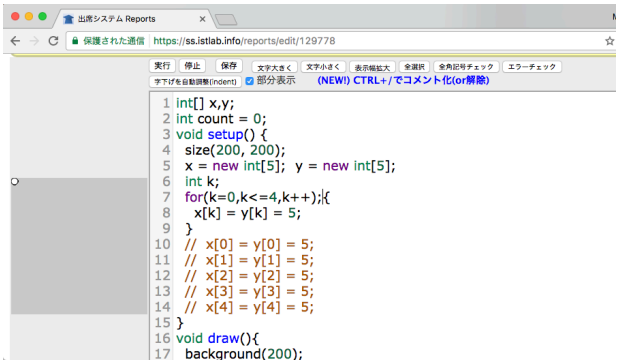

図 1 学習者が入力したエラーを含むプログラム

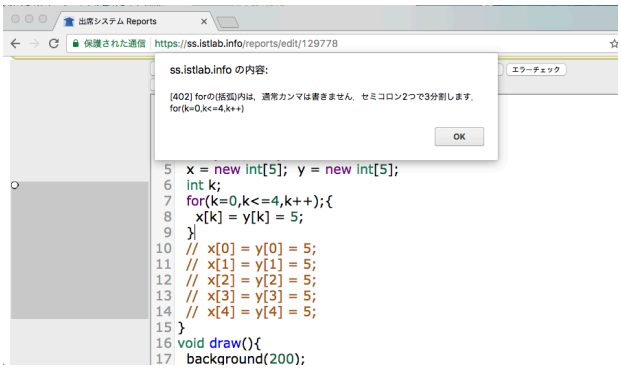

図 2 実行前無限ループチェックによるメッセージ

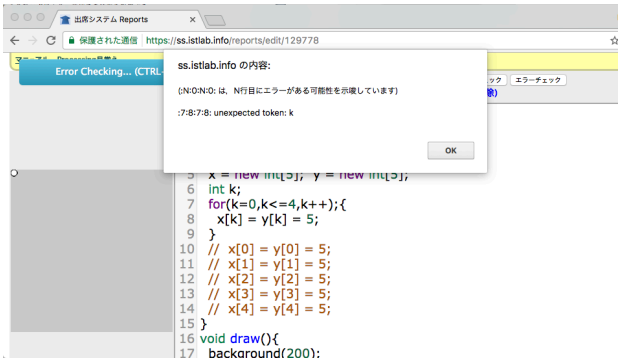

図 3 サーバ側コンパイルチェック結果（ループ）

図 4 は変数定義を忘れていた場合のスタンドアロン版 Processing のコンパイラによるメッセージである．Processing.js のコンパイラは，変数の定義忘れを指摘せず実行してしまう．複数のチェック機能を用いることで，学習者がエラーの原因を発見・修正し，学習の継続を促す効果が期待できる．

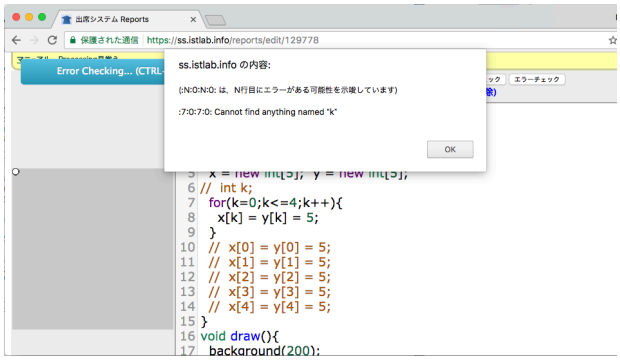

図 4 サーバ側コンパイルチェック結果（未定義変数）

提案システムを 2016 年 6 月 20 日から 2017 年 5 月 25 日まで，7 つの講義で利用した．その結果，511 回の無限ループ実行を回避することができた．試験や小テストのように，解答時間が制限されている場面において，ミスによる無限ループを事前に回避できることは，とくに効果的であることを確認した．

また，エラーの種類について分析した結果，本来小文字で記述すべき変数名を大文字で入力してしまうケースが多いことがわかった．この理由として，セミコロン入力時の SHIFT キーを離す前に，変数名をタイプしてしまっている可能性が考えられる．

5. おわりに

Processing.js を利用した Web IDE において，無限ループやエラーをチェックする仕組みを導入した．特に無限ループチェックは，初学者がタイプミスによるブラウザのフリーズを心配せずに，安心して演習に取り組めるという点で，重要であると考えている．実際の講義で運用した結果，特定のエラーが多いことがわかった．今後は，学習者がエラーをどのように解釈し，解消しようとしているかを調査することで，自発的なエラー原因の解消を促し，継続的な自主学習を支援していきたいと考えている．

謝辞：本研究の一部は JSPS 科研費（課題番号 15K00485）の支援によるものです．

参考文献

(1) Max Goldman, Greg Little, and Robert C Miller: “Real-time collaborative coding in a web IDE”, Proceedings of the 24th annual ACM symposium on User interface software and technology, pp. 155–164 (2011)
(2) Vu Nguyen, Hai H Dang, Kha N Do, and Thu D Tran: “Learning and Practicing Object-Oriented Programming Using a Collaborative Web-based IDE”, Frontiers in Education Conference (FIE), pp. 1–9 (2014)
(3) 三浦 元喜：“Processing Web IDE を用いたプログラミング基礎教育の試み”，情報教育シンポジウム 2013 論文集，pp.225–231（2013）
(4) 三浦 元喜：“Web 技術を活用したインタラクティブな情報教育環境の構築と実践”，電子情報通信学会信学技報 IEICE-ET2016-11 ，Vol. IEICE-116, No. 85, 名古屋工業大学，pp. 19-24 (2016)