

プログラミング演習システム PROPEL の Java 対応とエラーメッセージの改善

戸上稔崇^{*1}・北英彦^{*1}

Email: togami@ce.elec.mie-u.ac.jp

*1: 三重大学工学研究科電気電子工学専攻

◎Key Words プログラミング演習, Java, 演習システム

1. はじめに

近年の情報化社会においてプログラミングの学習はとて重要な位置にある。三重大学工学部電気電子工学科を含む多くの大学ではプログラムを実際に作成してプログラミング技術を身につけるため、また、計算機への理解を深めるためにプログラミング演習が行われている⁽¹⁾。

三重大電気電子工学科ではこれまで「プログラミング演習 I」および「プログラミング演習 II」の講義で C 言語を学習してきたが、今年度より Java の学習へと切り替えた。Java は広く使われているオブジェクト指向プログラミング言語であり、プログラムの再利用性が高く、分担しての作業が容易になるため大規模な開発での開発効率が高いとされている⁽²⁾。近年ではソフトウェア開発を Java に切り替えているところも増えてきており、Java への注目が高まってきている。

2. Java の学習環境

プログラミング演習における Java の開発環境として以下の3つを検討した。

- Eclipse, NetBeans, IntelliJ IDEA などの統合開発環境 (IDE) ⁽³⁾⁽⁴⁾⁽⁵⁾
- Java Development Kit (JDK) + ファイル共有システム⁽⁶⁾
- プログラミング演習システム PROPEL ⁽⁷⁾⁽⁸⁾

今回の研究ではプログラミング初心者をターゲットと定め、講師がリアルタイムで学習者全体のプログラム作成状況を把握できる必要があると考えた。具体的には、以下のことを講師ができることが必要であると考えられる。

- 作成途中のプログラムの閲覧
- プログラム作成に行き詰まっている学生の発見

Eclipse のプラグイン機能を用いた、学習者のプログラミング作成履歴を記録し、講師が開発過程を再生できるようにした先行研究⁽⁹⁾がある。しかし、学習者全体の作成状況を把握する機能はない。

JDK とファイル共有システムを使えば、講師は学習者のファイルを直接見ることはできる。しかし、同時に多くの学習者の状況を演習中に把握することはできない。

当研究室が開発している PROPEL (PROgramming Practice Easy for Learners) では講師が学習者全体のプログラム作成状況を把握できる後述のプログラム作成履

歴一覧表示機能および座席表示機能を持つ。PROPEL はこれまでは C 言語での学習を対象としていたためそのままでは Java への対応はしていなかった。

本研究ではプログラミング学習者への迅速で適切なアドバイスと、講師が瞬時にすべての学習者の状況を把握できるようにするシステムを講義に取り入れるため、PROPEL を Java に対応させることにした。

また、コンパイルエラーメッセージは専門的な用語が含まれているため初心者にはわかりにくい。そこでコンパイルエラー時に表示される文章を追加することで学習者がコンパイルエラーを取り除きやすくする。

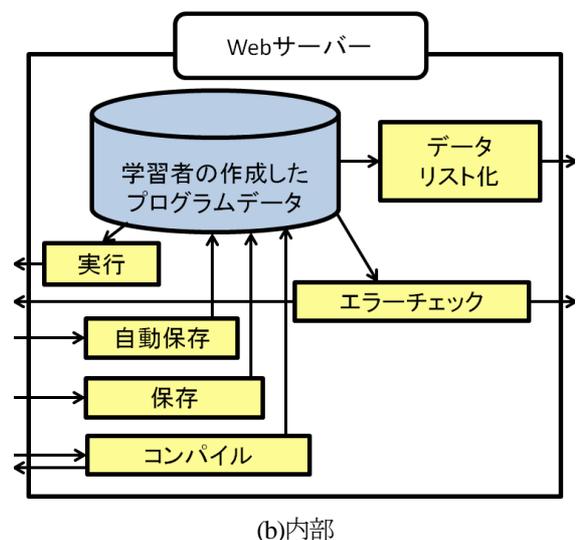
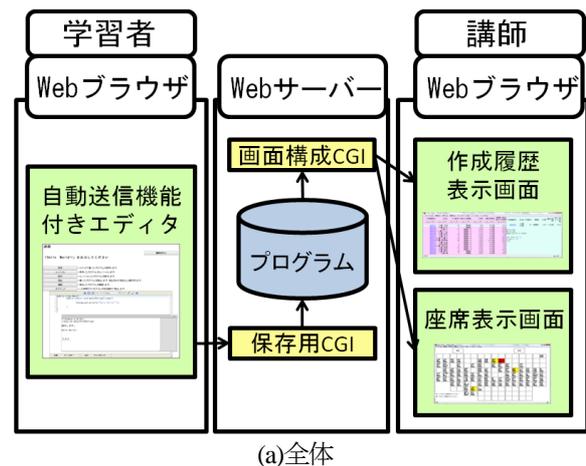


図 1 PROPEL のシステム構成

3. プログラミング演習システム

当研究室では、プログラミング演習支援を目的としたシステムを開発・研究してきた。そのうちの1つが学習者のプログラミング作成状況の把握及び、理解の遅れている学習者への迅速な対処を目的とした、プログラミング演習システム PROPEL である。PROPEL は、学生がプログラム作成をするための学生用画面と、学生のプログラム作成状況を把握するための講師用画面、データを管理する Web サーバーからなる。学生用画面と講師用画面は Web ブラウザ上で開くことができ、初心者を対象とした学習であれば、機能が多い統合開発環境を用いるよりも手軽にプログラムを作成することができる。PROPEL のシステム構成図を図 1 に示す。

図 1 (a) に示すように、学習者はウェブブラウザ上で動くエディタを使用する。エディタには自動送信機能があり、30 秒毎に作成途中のコードをサーバーに自動的に送信、保存する。保存された情報はまとめられ、講師用画面の作成履歴表示や座席表示画面に送られる。

図 1 (b) に示すように、Web サーバー内部では、自動保存や保存、コンパイルされたプログラムデータが記録されそれらのデータをまとめ講師に情報を提供したり、作成途中のプログラムに対するエラーチェック機能で学習者にプログラム作成段階でミスをしていないかなどの情報を返したりする。

学習者用の画面を図 2 に示す。学習者はデバッグを除くプログラミングにおける主な機能、保存やコンパイル、実行といった機能を利用できる。また、「講師を呼ぶ」ボタンを押すことで後述の座席表示画面に講師のアドバイスを求めているということが表示され講師を呼ぶことができる。



図 2 学習者画面

3.1 プログラム作成履歴一覧画面

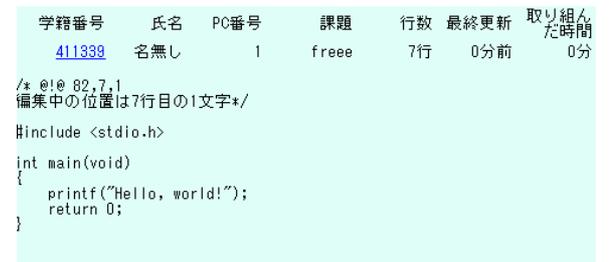
講師用の画面ではプログラムの作成状況を一覧で見ることができる。図 3 (b) では PROPEL で課題を行っている学生の学籍番号と名前、現在の課題、課題に取り組んだ時間などが表示される。学習者の作成中のソースコードを見たい場合には、見たい学習者の学籍番号をクリックすることで図 3 (c) のようにソースコードが表示される。この画面を見ることで、講師は誰がどの課題を何分間取り組んでおり、プログラムがどこまで作成できているのかなどをひと目で把握することができる。



(a) 全体図

学籍番号	氏名	PC番号	現在の課題	行数	最終更新	課題に取り組んだ時間
414328	坂田	0	kada14	65行	129分前	129分
414329	鈴木	0	kada12	50行	141分前	提出済み
411339	戸上	128	kada14	65行	123分前	提出済み
414334	竹内	0	kada12	30行	132分前	181分
414320	川原	0	kada12	44行	132分前	184分
414314	大中	0	kada14	65行	128分前	128分
414325	能谷	0	kada12	29行	134分前	185分
414304	伊坂	0	kada12	94行	133分前	提出済み
414322	喜田	0	kada12	74行	138分前	提出済み
414310	上田	0	kada12	36行	140分前	186分
414337	武田	0	kada12	35行	132分前	187分
414311	江崎	0	kada12	32行	136分前	179分
414306	石黒	0	kada12	41行	132分前	提出済み
414340	辻井	0	kada12	40行	140分前	提出済み
414302	浅岡	0	kada13	39行	136分前	提出済み
414335	竹岡	0	kada12	47行	132分前	189分
414326	栗野	0	kada12	29行	135分前	186分
411332	高崎	0	kada12	34行	133分前	提出済み
414317	小倉	0	kada12	39行	132分前	193分

(b) 学習者一覧表示



(c) 履歴表示

図 3 作成履歴一覧表示画面

3.2 座席表示機能

図 4 のように演習室の座席一覧が表示され、プログラムの作成に行き詰まっている学習者の座席の位置を知ることができる⁽¹⁰⁾。座席のマスには学習者の名前と作成中の課題名が表示され、誰かどこにいてどの課題を行っているのかが瞬時にわかる。座席のマスの中の色は、5 分以上更新がなければ黄、15 分以上更新がなければ橙、講師を呼ぶボタンを押していれば赤となり、作成の滞っている学習者を瞬時に見つけることができる。

PROPEL を用いることで、プログラミング演習において、プログラムの作成に行き詰まっている学習者を早い段階で見つけ出し講師がアドバイスすることが可能となる。また講師も、どのくらいの学生が行き詰まっているのか、また誰が講師の助けを必要としているかなど学習者全体の理解度を容易に把握することができる。

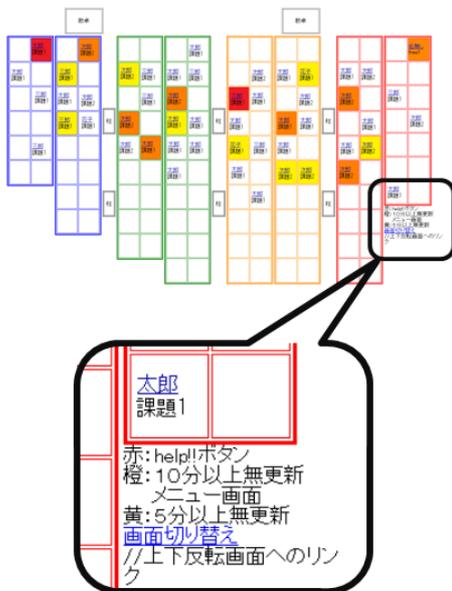


図 4 座席表示画面

4. PROPEL の Java への対応

先に述べたように PROPEL は C 言語での学習を前提としていた。そのため、これまでのままでは Java の学習で PROPEL を用いることはできなかった。Java に対応させ、プログラミングできるようにするにはいくつかの点を変更する必要がある。

4.1 Java でのコンパイル及び実行

図 5 で示すように、C 言語はコンパイル後マシンの上で直接実行しているが、Java はコンパイルによって Java バイトコードと呼ばれる中間言語コードへと変換し、それを仮想マシン上で実行する。図 1 (b) の「コンパイル」及び「実行」モジュールで、コンパイルコマンドと実行コマンドの 2 箇所を Java のもの書き換えることで、Java で記述されたソースコードのコンパイルおよび実行を可能にした。

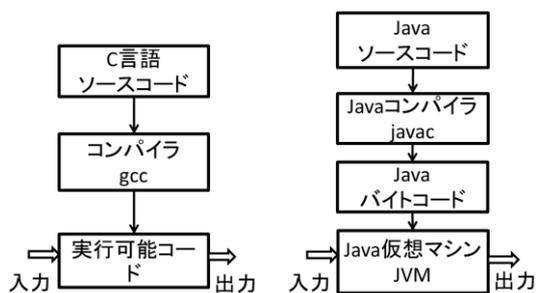


図 5 C 言語と Java のコンパイルの違い

4.2 プログラム作成履歴閲覧機能

作成中のプログラムの 30 秒毎の自動保存、保存ボタンによる保存、ギブアップ時、プログラム提出時のソースコードはすべて拡張子 c で保存されていた。これらを拡張子 java で保存し、読み取り時にも拡張子 java のファイルを指定することでプログラム作成履歴の Java への対応を可能にした。これらは図 1 (b) では「自動保存」「保存」「データリスト化」モジュールにあたる。

4.3 作成途中でのエラーチェック機能

当研究室の小島佑介が行った、2010 年のプログラミング演習 I の受講者を対象にした初心者がどのような間違いをするかの調査⁽¹¹⁾では次のような間違いが多く見られた。

- 「#include <stdio.h>」の記述忘れ
- 全角空白の混入
- セミコロン忘れ
- クラス宣言忘れ等価演算子「==」と代入演算子「=」の間違い
- クラス名、または変数名のタイプミス
- ダブルコーテーションのつけ忘れ
- “{”と“}”の間違い
- 括弧のとじ忘れ

小島の研究では、学習者が上記のプログラミングミスを起こしていた場合検出しエラーメッセージを表示するようになっている。「#include <stdio.h>」の検出を取り除いた他のエラーに関しては C 言語と Java で共通の性質であるため変更する必要はない。

4.4 コンパイルエラーメッセージ

学習者が自分でプログラムのミスに気付くにはコンパイル時のエラーメッセージを適切に読み取らなければならない。しかし、コンパイルエラーの中には分かりにくい文のものがあるため、プログラミング初心者はエラーメッセージの意味がわからず間違いを修正できない可能性がある。そこで、コンパイルエラーメッセージが表示された時にその理解の手助けとなる文が同時に表示されるようにする。今回は 3.3 で述べた調査で得られたプログラミングミスによるコンパイルエラーメッセージを対象とする。

これらの間違いにより表示されるコンパイルエラーメッセージと追加表示するメッセージの文を表 1 にまとめる。コンパイルエラーメッセージの中には「¥12288 は不正な文字です」「互換性のない型」など初心者にはわかりにくい表現が含まれている。文章を追加することで、プログラミング初心者でも自力でプログラムの間違いを修正できるよう促す。

学習者画面では図 6 のように、コンパイルエラーが発生した時に Java コンパイラから返されたメッセージの後ろに追加文章が表示される。

5. 今後の課題

今回、コンパイル時のエラーメッセージに文章を追加することで学習者の自力でのコンパイルエラー解決を促した。今回の変更で学習者の演習効率がどの程度変化したのか、今後の調査で検証する必要がある。

また、現在 PROPEL は 1 ファイルからなるプログラムの作成にしか対応していない。しかし、Java をより深く学ぶにはオブジェクト指向による複数ファイル、複数クラスからなるプログラムの作成にも対応するべきである。図 7 に対応した時の画面構成の予想図を示す。

表 1 対応するエラー原因

Java のコンパイルエラーメッセージ	学習者に提供する追加表示する文章
¥12288 は不正な文字です	全角空白が混入しています。特に、字下げの部分に全角空白がないか確かめてください。
エラー: ¥xxxxx は不正な文字です (xxxxx には数値が入ります)	文字列の内部以外の部分に全角文字が混入しています。例えば「;」「 」「 」「 」「 」などが全角記号「;」「 」「 」「 」になっていませんか。
‘;’がありません	式の終わりにセミコロン「;」をつけ忘れていました。
シンボルを見つけれません	名前 (シンボル) が定義されていないことを意味しています。変数を定義していないか、変数名のスペルミスの可能性があります。
文字列リテラルが閉じられていません	ダブルコーテーション「"」をつけ忘れていました。
互換性のない型	等価演算子「==」と代入演算子「=」を間違えていませんか。
)がありません	括弧を閉じ忘れている可能性があります。
変数 x は初期化されていない可能性があります	使用している変数に値が入力されていない可能性があります。
構文解析中にファイルの終わりに移行しました	文字列やブロックや括弧を閉じ忘れていませんか。

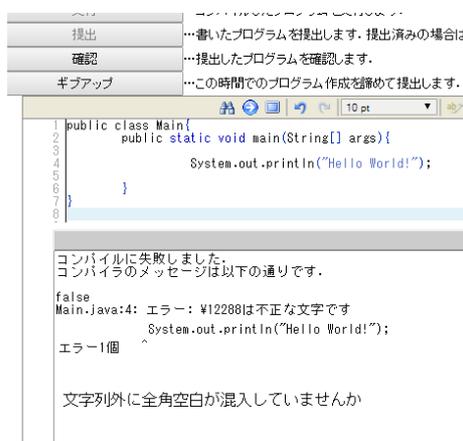


図 6 エラーメッセージの追加表示

課題

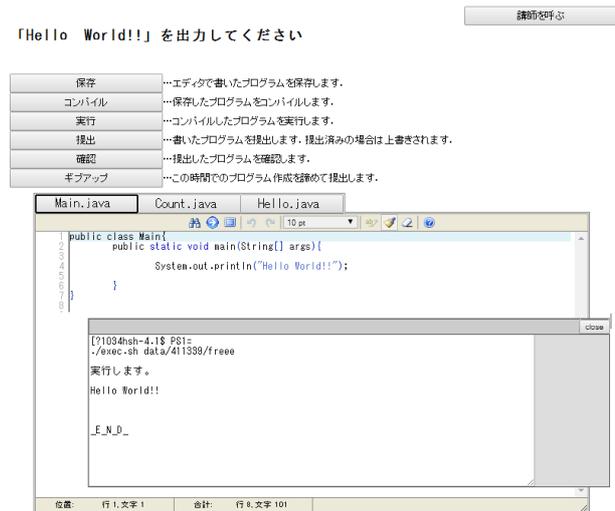


図 7 複数ファイルの作成への対応

6. おわりに

今年度より始まった Java でのプログラミング演習で学習者に対して効率よく支援ができるようにするために PROPEL を Java に対応させた。また、Java のコンパイルエラーを学習者が自力で修正できるように、コンパイルエラーメッセージに文章を追加した。PROPEL は現在、実際にプログラミング演習の講義で運用中である。PROPEL を利用することで従来よりもプログラミングを効率よく学習することが期待できる。

参考文献

- (1) 三重大学平成 26 年度ウェブシラバス: <https://syllabus.mie-u.ac.jp/> (参照 2015-2-24)
- (2) 粉名内仁章, 西智也: “新版オブジェクト指向プログラミング—J2SE1.4/5.0 対応 (Java バイブルテキストシリーズ)”, SCC (2007).
- (3) 石黒尚久, テクニカル書籍編集部: “Eclipse4.4 完全攻略”, SBCreative (2014).
- (4) NetBeans : <https://ja.netbeans.org/> (参照 2015-2-24)
- (5) IntelliJ IDEA : <https://www.jetbrains.com/idea/> (参照 2015-2-24)
- (6) Oracle Java SE Development Kit 8 : <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html> (参照 2015-2-24)
- (7) 森一樹, 田中昂文, 橋浦弘明, 樋山淳雄, 古宮誠一: “プログラミング演習支援のための細粒度履歴収集環境の開発”, 雑誌名, 巻, 号, pp.27-38 (2003).
- (8) 伊富昌幸, 北英彦, 高瀬治彦, 林照峯: “プログラミング演習のための初心者向け支援システム”, 雑誌名, 巻, 号, pp.27-38 (2003).
- (9) 伊富昌幸, 小島佑介, 高橋功欣, 北英彦: “プログラムの作成状況を把握する機能を持つプログラミング演習システム”, 雑誌名, 巻, 号, pp.27-38 (2003).
- (10) 小川正, 西口大亮, 北英彦: “プログラミング演習における iPad などの携帯デバイスの利用による指導の円滑化”, 雑誌名, 巻, 号, pp.27-38 (2003).
- (11) 小島佑介, 高橋功欣, 北英彦: “プログラミング演習における効率のよい指導のためのエラー早期指摘”, 雑誌名, 巻, 号, pp.27-38 (2003).