

LN-007

Java プログラミング演習向け課題レポート提出・管理機能を付加した授業支援システム Java Programming Exercises Management System in Class Support Environment

熱田 智士†
Satoshi Atsuta

松浦 佐江子‡
Saeko Matsuura

1. はじめに

近年、大学の情報系学部におけるプログラミング教育科目の題材として従来までの C 言語に加え、Java を取り扱うケースが多く見受けられる。これは Java がエンタープライズシステムにおけるサーバーサイドの実装言語としての実績を挙げていることや、多くの大学が Unix ベースの環境と Windows を併用していることから演習のためにプラットフォーム非依存の開発言語を求めていることに起因すると考える。また、Java は UML などモデリング言語との親和性の高さからソフトウェアの設計に関する議論においても取り上げられることが多く、学習教材としてメリットの高い言語であると言える。

Java 導入の流れに関しては本学も例外ではなく、2年前から Java 言語によるプログラミング演習を取り入れている。演習科目は履修者数が比較的多く、また課題レポートの提出回数も多いため、これらを効率的に提出・管理する必要性が当然生じた。

プログラミングのレポートは管理面での容易性も然ることながら、プログラムの正当性の検証を容易に行う上で電子媒体であることが望ましい。ただ、一口にプログラムのレポートといっても、扱うプログラミング言語によってその特徴は異なり、全てを一元的に扱うことは難しく、各言語に特化した支援が必要になる場合がある。

よって、本稿では、この Java によるプログラミング演習科目におけるレポートの電子的な提出及び管理を行うために開発したシステムの概要と運用結果について報告する。

2. システムの運用対象科目と要件

本システムの利用対象となった科目は「プログラミング演習 II」である。この科目はオブジェクト指向言語に関する講義科目と関連し、学部 2 年生を対象に Java による基本的なプログラミング技術の習得と、オブジェクト指向による設計・開発の概念を理解することを目的としている。2003年度の履修者数は140人、課題は毎授業ごとに課され、計13回実施される。内、7回が採点の対象となり、更にこれらは2通りの採点方式によって採点される。

2通りの採点方式の内、1つ目は単に提出されたプログラムのソースコードに対し、教員がその良し悪しを判断する方式である。この方法では、プログラムに対する理解力を判定し難いため、次のもう1つの採点方式を導入している。

2つ目の採点方法は提出されたプログラムの良し悪しを教員がソースコードのみで判断するのではなく、履修者の自らが記述したプログラムに対する理解力を測るために、実際に提出したプログラムをコンパイル・実行した後にアルゴリズムや処理の流れなどに関する説明を行って貰い、その内容によって採点を行う方式である（以下、この採点方式を審査方式と呼称する）。前者は採点対象となる7回の課題の内、4回実施され、後者の採点方式は3回実施される。

以上が授業の概要であり、この授業においてシステムに求め

られる主要な要件は以下の3つである。

(1) ソースコードのコンパイル結果の表示

履修者の中にはコンパイルが通らない不完全なプログラムをそのまま提出する者がいる。それらを篩いにかけてコンパイル結果の表示機能が求められる。

(2) Java のパッケージを利用したソースコードを扱える

Java にはパッケージと呼ばれる概念が存在する。この概念は簡単に言えば、Java のクラスを分類するための機能であり、C++言語における名前空間に近い振る舞いをする。なお、この機能はファイル・ディレクトリ構造に依存する。この概念を理解することは、オブジェクト指向開発を学ぶ上で重要であり、授業では実際にパッケージを利用した演習を行う。

(3) 自身の提出したレポートの取得が可能

審査方式による採点は授業時間中に実施され、効率良く採点を行うために、複数人の採点者が各学生の席を回りながら採点を行う。よって、採点の際には提出されたレポートを提出者に返却しなくてはならない。これは採点に公平を期すため、説明の対象となるプログラムはあくまでも指定したレポート提出期間内に提出されたものを用いるためである。

以上3点の要件を満たすことを軸とし、システムを実装した。

3. Java プログラミング演習科目のレポート形式

3.1. プログラミング演習科目に適したレポート媒体

レポートは従来から今日に至るまで、紙媒体による形式が広く用いられているが、ことプログラミング演習科目のレポートに関しては、適切な選択肢ではない。何故ならば、プログラミング演習のレポートの内容とは概ねプログラムそのものであり、それらを評価するためにはプログラムの正当性を検証しなければならない。もしレポートの提出が紙により行われたとしたら、プログラムを読み解くか、もしくは実際にプログラムを全て入力し、実行する以外で正当性を評価することは難しく、そのような方法は現実的でない。またプログラムが一定以上の規模になれば、使用される紙量も膨大と成る。レポートの管理方法や、昨今の森林問題なども考慮すれば、この問題を軽視することはおそらくできない。以上の点から、プログラミング演習科目のレポートに紙媒体を選択することは不適切であると言える。プログラミングの演習科目に最も適した媒体は電子媒体であると考え。レポートの提出を電子メールなどで行う方法は一般的に成りつつあるが、プログラムのレポートの場合、管理面でのメリットも然ることながら、評価の際にプログラムの正当性の検証が比較的容易になることが大きい。これは間にアナログ媒体を介さないため、プログラムを実際に実行したりすることが容易に行えるためである。故に、プログラミング演習科目のレポートの提出・管理は電子的であることが望ましいと言える。

3.2. プログラミング言語によるレポート形式の違い

レポートを電子的に扱うといっても全てを一元的に扱えるわけではない。レポートの電子的に扱う際の、最も一般的な形式はおそらく単一の文書ファイルである。これは紙媒体によるレポートをそのまま電子化したと捉える事ができるため、広く適応する方式であると考えられる。しかし、この方式はプログラ

† 芝浦工業大学 (SIT) 大学院

工学研究科電気電子情報工学専攻

‡ 芝浦工業大学 (SIT) システム工学部電子情報システム学科

ミング演習科目には必ずしも適応しない。プログラムのソースコードは一般に複数のファイルで構成されるため、これを単一の文章ファイルで表現しようとする、複数のソースコードを1つに纏めなければならない。そして提出されたソースコードを検証するためには、文章ファイルから何らかの方法—おそらく専用のツールなどを用いて—ソースコードを分割・抽出する必要がある。

この方式はC言語の演習科目である「プログラミング演習I」で用いられていた。ここで、その際に利用したシステムの概要について述べる。

・C言語演習用レポート提出システムとレポート形式

このシステムでは、複数のソースコードと実行結果をひとつのファイルに纏め、メールの本文に貼り付けて提出を行う。その際にファイルは各ソースコードと実行結果の境目に区切りとなる文字列「%%題名%%」を挿入した独自のフォーマットに従う。題名は教員側から予め指定しておき、提出されたファイルはシステム側で区切り文字及び、区切り文字の題名を元にして再度分割・コンパイルされ、その結果を閲覧することができた。なお、ファイルを分割する際に分割されたファイルは同一のディレクトリに展開され、階層構造を持たない。このように単一の文章ファイルに独自のフォーマットを用いたシステムは他にも存在している[1]。

この方式は学生にファイルフォーマット強制するという欠点を伴うが、C言語の演習科目においては概ね上手く機能した。本システムの対象となるJavaの演習科目も開始当初は、上記ツールを用いて課題レポートの収集管理を行っていた。しかし、C言語の演習を対象として作成されたツールには、Javaのソースコードを扱う上で不都合な点が見受けられた。それは上記ツールが分割したソースコードを全て同一ディレクトリに展開してしまう点である。C言語のソースコードは階層構造を利用する必要性が薄いため、ディレクトリ構造を持たなくともこれといった問題はなかったが、本システムの要件(2)より、システムはパッケージを扱えなければならない。先ほどの述べた通りJavaのパッケージの概念はディレクトリの階層構造に依存しているため、この方式では、Javaのパッケージの概念を表現することができない。また、本システムの要件(1)により、提出されたソースコードはコンパイル可能な状態でなければならない。パッケージを利用したソースコードは階層構造を保持したままでは、コンパイルを行うことができないため、システムには提出されたソースコードの階層構造を保持することが求められる。

このようにプログラミング言語のレポートに限っても、扱うプログラミング言語の特徴によって最適なレポートの形式は異なる。

3.3. 本システムのレポート形式

よって、本システムではJavaの特徴を踏まえ、ディレクトリの階層構造を保持する課題提出方式を提供する。その際に、従来のツールのような特殊フォーマットを持つ単一のドキュメント形式から脱却するために、複数のファイルから構成されるレポートの提出をサポートする。提出時に動的にディレクトリ構造を構築することでこれを実現する。

4. システム

本システムは、表題にあるとおりJavaの課題レポートの提出・管理のみを行うシステムではない。本システムは非リアルタイム・分散型のシステムであり、授業に関して学生・教員双

方の観点から総合的に支援することを目的としている。レポート提出・管理機能は本システムの機能の一部として実装されている。

このような形で実装されているのは、本システムの適用対象をこの科目に限定せず、いずれ他の科目での利用を視野に入れているためである。なお、本システムの仮称は「Class Support Environment」の頭文字を取り「C.S.E」としている。システムは現時点で授業を支援するためにレポート提出機能以外に以下の機能を備えている。

表1. システムの主要機能

	機能
学生側	シラバスの閲覧 連絡事項(教室変更・休講情報)の閲覧 教材(講義・課題教材)のダウンロード アンケートの回答 授業に関連するリンク閲覧 教員プロフィールの閲覧
教員側	シラバスの編集 連絡事項の追加・編集 教材のアップロード アンケートの作成・集計 授業に関連するリンクの追加・編集 履修者の登録・削除 プロフィールの編集
管理者側	学生の追加・削除 教員の追加・削除

4.1. 構成

本システムはWebブラウザをインターフェースとするWebアプリケーションであり、サーバーサイドはASP.NETによって実装されている。動作環境はOSにMicrosoft Windows 2003 Server、WebサーバーとしてInternet Information Services (IIS) 6.0、データベースにはMicrosoft SQL Server 2000を採用した。マシンスペックはCPU—Celeron 2.0Ghz、メモリ—1GBであり、システムはこの単一のマシン上で全て実行される。

クライアントサイドはWebブラウザとしてInternet Explorer (IE) 6.0を推奨環境とし、またMozilla 1.0も動作対象とした。

4.2. 機能とインターフェース

シラバス、連絡事項、教材などを含めた総合的な授業支援機能については、他の機会に別途報告する予定である。以下では課題レポートの提出・管理機能についてののみ簡単に説明する。

4.2.1. レポート提出

本システムにおける課題は1つ以上の問題から構成される。そして、各問題はその提出形式の違いによって、大きく分けて2つに分類される。

1つは単一ファイル形式である。この方式は、課題レポートが単一のファイルで構成される場合に用いる方式である。これは一般的な提出方式であり、ほとんどの課題はこの方式を採用と思われる。

もう1つは複数ファイル形式である。この方式は本システムの要の機能の一つであり、システム要件(2)を満たすためにディレクトリの階層構造を保持したソースコードの提出を実現する方式である。

・課題の提出方法

レポートは、教員が指定した期間内のみ提出可能であり、締め切り日時を過ぎると即座に提出操作が禁止される。

レポートの提出は問題の形式が単一ファイル形式であった場合には、単にアップロードするファイルを選択してアップロー

ドボタンをクリックするだけで良い。

複数ファイル形式の場合は、Web ブラウザ上で、ディレクトリ構造を構築しながら提出を行う。図1が実際の提出画面である。画面左側のツリーがこの問題に対して構築されたディレクトリ構造を表している。ノードを選択することで選択されたディレクトリに移動することができる。

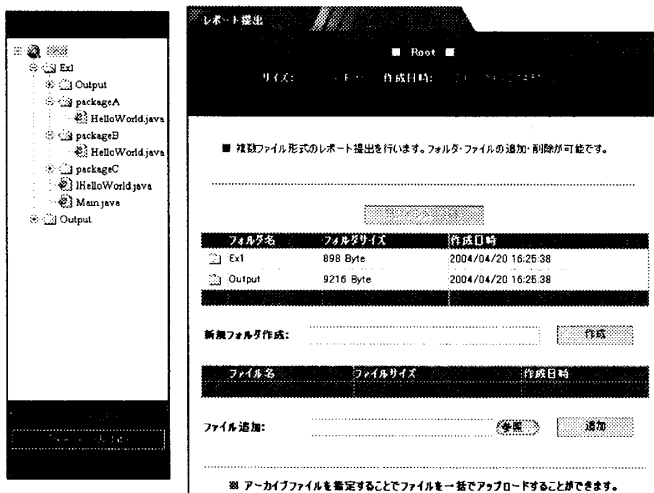


図1 複数ファイル形式レポート提出

ファイルのアップロードは基本的にはアップロードしたいディレクトリに移動した後、ファイルを1つ1つ選択し、アップロードを行う。http の制約により同時に複数のファイルをアップロードすることはできない。この方式の利点は、提出時にディレクトリ構造を構築することができる点である。よって、ローカルに提出用のディレクトリ構造を事前に構築しておく必要はない。

これに対して事前にローカルに提出用のディレクトリ構造を構築している場合、1つ1つファイルをアップロードするのは、冗長な操作である。この問題を解決するためにアーカイブされたファイルをアップロードし、そのファイルをサーバーサイドで展開してディレクトリ構造を構築する機能を実装している。

なお、1つ1つアップロードする方法だと当然ながら構成されるファイル数が多くなるのに比例して作業量が増大する。よってファイル数がある程度多くなった場合には後者の方法を利用すべきである。

・Java ソースコードに関するオプションの指定

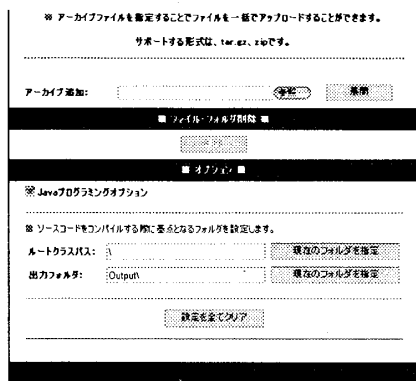


図2. Java オプション指定

ソースコードのアップロードが完了したら、最後にソースコードのコンパイル時に基点となるディレクトリと出力ディレクトリを指定する(図2)。

この指定はシステムによって強制される訳ではないが、教員側でコンパイル結果を閲覧する上で必須となる項目である。

4.2.2. 提出されたレポートのダウンロード

・学生側： 学生は自分が提出したレポートをダウンロードすることができる。その際、問題ごとにダウンロードするか、課題全体を纏めてダウンロードするか選択することが可能である。この機能は、審査方式による採点の実施を支援することを想定して実装した機能である。

また、学生が自身のレポートの提出状況を確認できることは、学生の観点からすれば必要な要件であると言える。課題をダウンロードすることで、レポートが正しく提出できているかどうかを確認することが可能である。

・教員側： 教員がレポートを採点する際に、ネット環境を常に要求されるのは都合が悪く、レポートをローカルで閲覧したいケースはあり得る。そこで、教員側では各課題について提出された課題を一括でダウンロードすることができる。

ダウンロードの際、課題はアーカイブに纏められるが、その際にアーカイブの形式は zip, tgz の二つをサポートしている。また、ディレクトリの構造についても二通りから選択することが可能である。1つは図3左側のような「問題→学生」という階層構造、もう1つは右側のような「学生→問題」といった階層構造である。前者は問題ごとに採点を行う場合に適した構造であり、後者は学生ごとに採点するのに適した構造である。

現時点では、ダウンロードされるアーカイブはサーバー対してポストバックが発生してから、サーバーサイドで圧縮処理を行うため、履修学生が多い場合などはサーバーに対する負荷が大きく、システムに対するレスポンスが著しく低下する可能性がある。

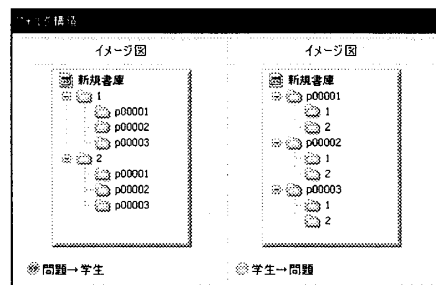


図3. レポートのダウンロード

4.2.3 提出された課題レポートの閲覧

これは教員側だけに設けられた機能であり、提出されたレポートを Web ブラウザ上で閲覧・評価することを目的とする。インターフェースは学生の複数形式のレポート提出の際のインターフェースと似通っている(図4)。

画面左側はディレクトリ構造を表すツリービューであり、クリックすることで該当するディレクトリに移動できる。

画面中央が簡易的なビューになっており、Java のソースコードもブラウザ上で表示することが可能である。またフォントサイズの指定(大中小)と、プログラムを作成した環境が

Unix 系と Window のどちらでも閲覧可能なように、文字コードの指定 (Shift JIS、EUC-JP) が可能である。本来ならば文字コードの自動判別のためのロジックを持つべきであるが、本システムでは手動で選択する必要がある。

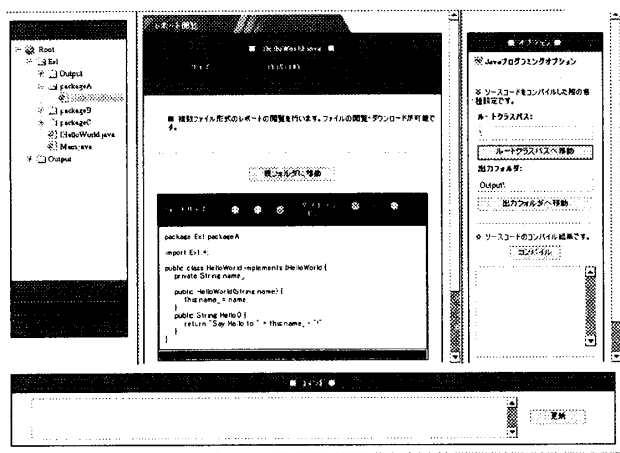


図4. レポートの閲覧・採点

画面右側の部分は学生が提出時に設定した Java のオプションが表示される。機能としてコンパイル時の基点となるディレクトリや出力先に移動することが可能である。また、ソースコードのコンパイル結果もこの部分に表示される。

画面下部はこの問題に対してコメントを記述するための領域である。このコメントは学生サイドから閲覧することが可能である。

4.2.4. レポートの採点

・学生側： 上記の通り、学生は教員がレポートに対して行ったコメントを閲覧することが可能である。

・教員側： 教員は課題の各問題について配点とコメントを設定することができる。教員は配点の範囲内で提出された課題に対して得点をつける。また、学生ごとの総得点を閲覧することが可能である。

5. 運用結果と考察

本システムの2004年4月現在の運用実績は「プログラミング演習Ⅱ」、グループワークによるソフトウェア設計・開発を目的とする科目「情報実験Ⅱ」において一部成果物提出、アンケート実施、及び本学システム工学部電子情報システム学科の卒業論文の概要の提出において運用した。

以上の運用に際し、運用に支障をきたすような致命的となるトラブルに見舞われることは無かった。以下で、各機能運用結果とそれに対する考察について述べる。

(1) ディレクトリ構造を構築するレポート提出方式はスムーズに受け入れられた。従来のように特殊なフォーマットに従わせるよりも、ディレクトリの階層構造を構築するという普段からやりなれた方式であるため、直感的に理解しやすかったのではないかと考える。また特殊フォーマットを取りやめたことで、提出されたレポートのフォーマットのエラーといった問題も取り除かれた。また、この提出機能は、「情報実験Ⅱ」においても利用された。この科目は半期をかけてソフトウェアの設計から開発までの全工程を行い、フェーズごとに成果物のドキュメントをレポートとして提出する。その際、成果物が複数ある場合に単一のファイルに強引に纏めるのではなく、この提出機能を用いて提出を行った。これは、この提出方式がプログラムの

レポート以外で複数のファイルから構成されるレポートを提出する上で有効であることを示す事例である。

(2) Java ソースコードのオプションを設定することは、教員側でコンパイル結果を閲覧する上で必要不可欠であるので、履修者には設定を徹底することを常に呼びかけた。それに関わらず、設定を怠るものがいた。これは履修者の不注意であると言えそれまでであるが、それ以上にこの設定を行うことは提出の際にシステム側から強制される項目ではなかったことが主な原因であると考えられる。

(3) レポートの一括ダウンロード機能は予想通りレスポンスの低下を招いた。この機能は頻繁に利用されるものではないため、システム運用上の問題にはならないが、それでも全ての処理をユーザーからのポストバックをトリガーにして行うのは設計上問題があったと言わざるを得ない。アーカイブの作成など一部高負荷の処理は今後、スケジューリングタスクなどに移行する必要がある。

(4) レポートの閲覧・採点画面 (図4) において、階層構造を辿りながらソースコードを閲覧できたことに関して教員からの感想は良好であった。また、コンパイル結果などが表示されている画面右側フレームが常に表示されている必要はないとの意見を得た。今後インターフェースの更なる改善が求められる。

(5) 提出期間を設定できることは、学生に提出期限を厳守させることに繋がった。以前のような紙によるレポート提出の場合、教員と学生、つまり人と人との間でレポート受け渡しが行われるため、ある程度の融通が利いた。しかし、間にシステムを挟むことで融通を利かすといったことは実質不可能になるため、時間を厳守するようになったと思われる。これを利点と見るか、欠点と見るかは教員より考えるか、学生より考えるかで意見の分かれる所であると思われる。

6. むすび

全ての科目に適応し、求められる要件を十分に満たすシステムを実現することは難しい。機能を一般化すれば全ての課目の最大公約数的なシステムになるが、個々の授業の形態に適したシステムには成り得ない。市販されているシステムは製品としての性格から一般化された機能で構成されている場合が多い。かといって、1つの授業に完全に特化してしまえば、適応できる科目が限りなく限定されることになる。よって、一般性をいかに損なわずに個々の授業に特化した機能を付加するかがこのような授業支援を目的とするシステムを設計するに際し、重要であると考えられる。運用結果で述べた Java のオプションの設定を強制しなかったことに対する理由は、一般性を損なうことを恐れたためである。ディレクトリの階層構造を構築するレポート提出方式は Java の演習科目に限らず他の科目にも適応する可能性がある。故に Java に特化した項目はシステム内で最小限に留め、他の科目に適用した際に利用者の操作の妨げとならないように配慮した。

今後、本システムのレポート提出機能は、特異な授業形態を持つ科目一例えば学生同士で互いのレポートを評価し合うことを行う科目などに適応するように機能を拡張していく予定である。また、本稿では詳しく述べなかったアンケートなどの他の機能についても今後改良を重ね、システム全体として学生・教員双方の観点から総合的に授業に関する支援を行っていきたいと考えている。

参考文献

[1] 石浦菜枝佐・他, "Cプログラミング演習のレポート投稿・管理システム", 2003, 第11回全国大学情報教育方法研究発表会