The Interactive Programming Portfolio

John K. Estell
Bluffton College

Introduction

The portfolio has long been used as a tool for monitoring student progress. In computer science, the programming portfolio contains a selection of computer programs that a student has produced over a period of time. Usually this has consisted of a notebook containing pages of program listings, perhaps combined with text-based example runs or graphics-based snapshots showing particular moments of program execution. While useful, reviewing such material is about as exciting as watching paint dry and fails to capture the essence of the programs in action. With the advent of the World Wide Web and the Java-based applet, the programming portfolio can be revitalized into an interactive format conducive to both student and reviewer.

The Portfolio

A portfolio consists of a collection of materials assembled over a period of time that is used to both demonstrate and document one's ability in a particular subject. Portfolios are commonly used in the artistic professions. For example, photographers who specialize in weddings will present to the inquiring engaged couple an assembled collection of their work. By constructing a portfolio photographers have the opportunity to reflect upon their work as they select the best results from their photographic sessions; similarly, the couple looking to hire someone for their wedding can use the portfolios to evaluate the ability of each photographer. So not only is the portfolio a means to demonstrate and document competence, it also allows for assessment by both the person assembling the portfolio and those who must pass judgement on that person's work. It is for these reasons that many in the engineering education community have adopted the use of portfolios in the classroom as it is a valuable assessment tool for both students and instructors.

With the rise of the World Wide Web there has been increased interest in the development of electronic portfolios. The nature of the Web as an interactive multimedia facility that can provide information on demand opens up new possibilities for the use of portfolios in many disciplines, including computer science. One variant recently reported on is the development of a Web-based on-line journal for their writing-intensive undergraduate software engineering course[1]. The on-line journal was used to allow students to conveniently maintain their journals while also providing the instructor a more effective way to review, and provide feedback on, the contents. Applying the concepts of an electronic portfolio to computer programming itself is straightforward. It is simple to create a Web document containing listings and descriptions of one's computer programs. The pages within this document may include graphical snapshots of programs "in action" and links to downloadable source code and executable files. The problem with this approach is the amount of effort required by the

reviewer to evaluate the material. It is worthwhile to download the source code only if one possesses a suitable compiler or interpreter for the language in which the program is written. Furthermore, despite the claims of the portable nature of high-level programming languages, a sufficiently robust program can easily contain platform-dependent code that would prevent its compilation and execution on different systems without some degree of modification. The downloading of an executable file also suffers from platform dependencies; in addition, there is an element of risk associated with running this type of file on one's own computer as the code could contain a virus.

Java Applets and Interactive Portfolios

The Java programming language provides a new design paradigm for programmers: write once, run anywhere. This object-oriented language was designed to maximize portability by specifically defining many of the details of the language for all implementations, from the source code all the way down to the byte code for the abstract machine language into which Java code is translated[2]. Java programs can be executed on any platform that has a Java Virtual Engine written for it; this engine is usually incorporated into all Web browsers. While Java programs can be written as stand-alone applications, they are best known in their applet format, which allow interactive programs to be incorporated into Web page design.

At Bluffton College, Java is the language used in the object-oriented programming (OOP) courses that our students take in their second year of study. Many of the non-OOP constructs of the language are derived from C, which is learned by our students in their first year of study. Accordingly, the introductory OOP course focuses on the OOP aspects of Java. In order to explore these concepts through the writing of programs, applets are introduced early in the course curriculum. This requires that students obtain a fundamental understanding regarding event-driven GUI programs. A minimal, but sufficient, amount of coverage is presented at this time regarding the use of Symantec's Visual Café Java development tool, the button, label, and text field GUI components, a small number of methods associated with these components, and how one handles action events generated by components. Once this material is covered students are able to write simple Java applets.

Now that the students have a basic understanding of event-driven GUI programming using applets, the interactive programming portfolio (IPP) can be introduced. The students are first instructed to create a Web document in their computer accounts that will serve as the home page for their portfolio. As each Java program is written, a new Web document is created to contain the HTML code required for a browser to invoke its Java Virtual Engine and execute the applet. This document also contains a copy of the source code for the applet being displayed. Encapsulating the source code within <PRE> tags so that the alignment of the code is preserved using the traditional fixed-width font is the best approach for incorporating the source code into the document. If the applet utilizes additional user-defined classes, each class is placed into its own file and a link to that file is incorporated into the Web document for that applet. Once finished, the student modifies the home page for the IPP to include a link to the Web document containing the new applet. At this point the applet is available for review by the instructor, who has the luxury of sitting at the computer, surfing to the appropriate Web page, and interact with the executing applet. The instructor can evaluate the program without having to fumble around

with a bunch of floppy disks or with scripting programs designed for handling electronic submissions, and without needing to compile source code. The IPP is an efficient and effective way for submitting Java applet programs for evaluation.

The IPP has been used in our OOP courses for students to demonstrate competence in OOP through a combination of self-designed and instructor-assigned applets. Students best learn programming by writing programs; the IPP is used to encourage them to explore aspects of the Java language on their own by providing a vehicle to showcase their ability. Due to the multimedia features incorporated into the language, many students gravitate towards exploring applications involving graphics and sound. Some of the student-designed applets that have been written include a graphing window, a calculator, and a dancing John Travolta with "disco fever." Games are also popular; some that have been written include skeet shooting, Mad Libs, and poker. Currently a couple of students are investigating speech synthesis applications in Java. Instructor-assigned applets are used both as an assessment tool and as a way to ensure that students write programs that incorporate certain language constructs. Some of the assignments that have been given include implementing the game of Yahtzee complete with graphically-displayed dice, creating classes derived from an abstract class for a shape manipulation program, and a drag racing simulation. This assignment paradigm has shown itself to be an effective motivational tool for most students. Students work on programming topics that are of interest to them. Because of this interest the average student writes mode code and places greater effort into ensuring its correctness than is normally encountered in the typical programming assignment. The IPP has also been used by the instructor as a means to make example programs available to the students in the class. This provides students with both the source code, which is also distributed in class as a handout when the program is discussed, and the working applet so that they can see the program in action.

Drawbacks and Potential Pitfalls

Using a Web-based programming portfolio does have its drawbacks. The greatest drawback for the IPP is that it can interactively showcase only those programs written in Java. If a student wants to include examples from other languages into the portfolio, the old static format of program listings and downloadable code must be used. It should be noted, however, that a Java applet displaying a looping sequence of snapshot images showing the operation of a program could be incorporated into the Web document for a program written in a language other than Java. Another potential problem is plagiarism. As noted in a recent article, there is an increasing number of "cybercheaters" who abuse electronic media by first using a Web search engine to easily sift through millions of on-line documents, then perform "cut and paste" operations from relevant sources to produce their "original" work[3]. There are many Java applet repositories already available on the Web that can be easily found using any Web search engine. Several of these sites provide the source code for the displayed applets. Additionally, there is the potential for visiting other student IPPs and copying the code presented there. Fortunately, the same tools that allow cybercheaters to operate can be used to verify suspicions of plagiarism. Fragments of suspicious code can be entered into a search engine in order to determine if it has been lifted from a listing available on the Web. One method that can be used to counteract some of the problems is to have students design their IPPs to allow full access to the applet, but hide

the source code files behind a password-protected link for the duration of the course. This would counteract copying off of other students currently in the course.

One of the problems always encountered in the classroom is the lack of motivation among the weaker students. If left to their own ways, these students will write few, if any, programs on their own. In an idealistic setting it would be nice to have students write their own programs to learn and explore the various aspects of programming as they progress through the course material. Unfortunately, the reality is that these students are often unsure as to what to do or are in a procrastination mode since there is no ominously approaching deadline staring them in the face. There are a couple of solutions to this problem. First is to incorporate periodic evaluations of student portfolios into the grading scheme; a firm deadline is a good motivational tool. To assist students in determining what programs to write, a mixture of traditional and flexible assignments can be given. The traditional assignment is where a problem is stated and the students are required to design and implement a solution. This type of assignment allows for assessment of the students via a common metric. A flexible assignment is best used to motivate students to exercise components of the language without stifling creativity. A simple example would be to ask students to write a Java applet that will dynamically modify the text of a displayed component. This could involve changing the text, the font type, style, or size that the text appears in, or the color of the text. By providing this information the struggling student now has some general guidance that is usually sufficient for getting started on a program.

Conclusion

The use of the interactive programming portfolio has shown itself to be an effective assessment tool for both the student and the instructor. The IPP has also shown itself to be useful as part of a resume; one former student received a job partially due to his employers being impressed with his Java programming skills, which were demonstrated by his IPP. While only programs written in Java are truly interactive when featured on a Web page, Java applets can be designed and developed to enhance the look of portfolio entries written in other languages. With the continuing rise in stature of Web-based documents, the interactive programming portfolio will play a greater role in documenting and demonstrating the competency and ability of computer programmers.

Bibliography

1. Riser, R. and Gotterbarn, D. On-line Journal: A Tool for Enhancing Student Journals. SIGCSE Bulletin (September 1998), 203-205.
2. Arnold, K. and Gosling, J. The Java Programming Language. Addison-Wesley, Reading, Massachusetts, 1996.
3. Ryan, J. J. C. H. Student Plagiarism in an Online World. ASEE Prism (December 1998), 20-24.

JOHN K. ESTELL
John K. Estell joined Bluffton College as an associate professor of computer science in 1996. He was previously an associate professor at The University of Toledo. He received a B.S. (1984) degree in computer science and engineering from Toledo and received both his M.S. (1987) and Ph.D. (1991) degrees in computer science from the

University of Illinois at Urbana-Champaign. His areas of interest include programming development tools and interface design. Dr. Estell is a member of ACM, ASEE, IEEE, Tau Beta Pi, and Eta Kappa Nu.