Eclipse Plugin Java Tutorial

Written by:
Stuart Fehr
Colorado School of Mines
sfehr@mines.edu

In association with:
Matthew Gimlin
Colorado School of Mines
mgimlin@mines.edu

Fall 2006

Abstract

In this paper, a new way to present tutorial material to a student is examined that removes unnecessary excise, increases student's attention at the concepts and materials, and provides immediate feedback to the user. By utilizing a plugin to the Eclipse development environment, all study materials and tutorial information can be integrated with the development environment in one window. This prevents the usual break in flow experienced by most students learning a new programming language with a standard tutorial.

1 Introduction

One of the major problems facing students in Computer Science programs and professional software developers is the challenge of adapting to and learning new programming languages and APIs. There are many tutorials for most programming languages and techniques available online. However, the non-interactive nature of these programming tutorials makes the material difficult for most students to absorb. In addition, many of these tutorials are written for a very very general audience in an attempt to make them as accessible as possible. For many reasons, this can be a frustration to students and professionals that are already well versed in another programming language or computer science principles.

Our project was the development of a new type of tutorial system that was specifically geared toward students that were already well versed in another programming language. In our case, we assumed that our students would be well versed in C++ and would be learning Java, although the design principles presented in this paper could be applied to many other programming language combinations. Our specific goal was to develop a new type of tutorial that would address many of the problems that we discovered with conventional static web-based programming language tutorials.

2 Related Work

There are many fine examples of tutorials available free of charge online to learn the Java language. One of these is the tutorial provided directly by Sun Microsystems [1], the distributors of Java. This tutorial is geared well toward a very wide audience of people that may want to learn the Java language. There are almost no gaps in the material that is covered by this tutorial, and it is available on their website, making it accessible from anywhere. In addition to good discussion of programming technique in Java and extensive example code to illustrate their concepts, the Java tutorial also includes informative graphics where appropriate and sometimes small Java applets to showcase some of the interesting things that can be done with Java programs.

There are also a myriad of books available on the Java language, too many in fact to discuss them in any depth. However, we found a book approach to be even more difficult to use than a web-based approach. The reason for this is that the flow of the students learning and programming process is repeatly broken by having to adjust between using a computer and navigating and using a reference book. In addition to the problem of having to constantly switch between the book and computer, unless the book came with a CD of source code there is no way for the student to compile and test that code directly, other than to tediously type in all of that code by hand. This is a very boring process which incurs no real learning for the student.

There have even been a couple attempts at making computer programming more accessible to students that would not normally learn a programming language. The most notable of these [2], used the idea of writing a novel to motivate the process of writing a computer program. However cute this may be, for a seasoned professional or an advanced student, this technique would be more annoying than useful for learning the Java language.

3 Design Philosophy

The main goals of our design were simple. First and foremost, we endeavored to never break the user's flow. This was the primary reason that we chose to create a plugin to the Eclipse environment. We felt that allowing the user to do all of their reading and experimenting without having to switch contexts was very important. It was also important that the user get instant feedback when they made a syntax error. Instant error highlighting was one of the compelling reasons to use Eclipse over any other text editing program that simply had syntax sensitive coloring.

In addition to never breaking the user's flow, we chose to make this program read simple HTML pages so that it could be generic and not require significant reprogramming if someone wanted to apply this plugin to a different kind of programming language such as C++. The Eclipse environment has recently been expanded to offer C++ development capabilities which could also benefit from a built in tutorial. This also helped to make our tutorial environment similar to the webbrowser environments that most users would already be comfortable in.

After observing students learning C++ in an introductory course at the Colorado School of Mines, we noticed that most people undertook a three step cycle when learning the C++ programming language. Each of these steps required switching contexts from one window to another, greatly breaking the user's flow and forcing them to concentrate on navigational tasks rather than

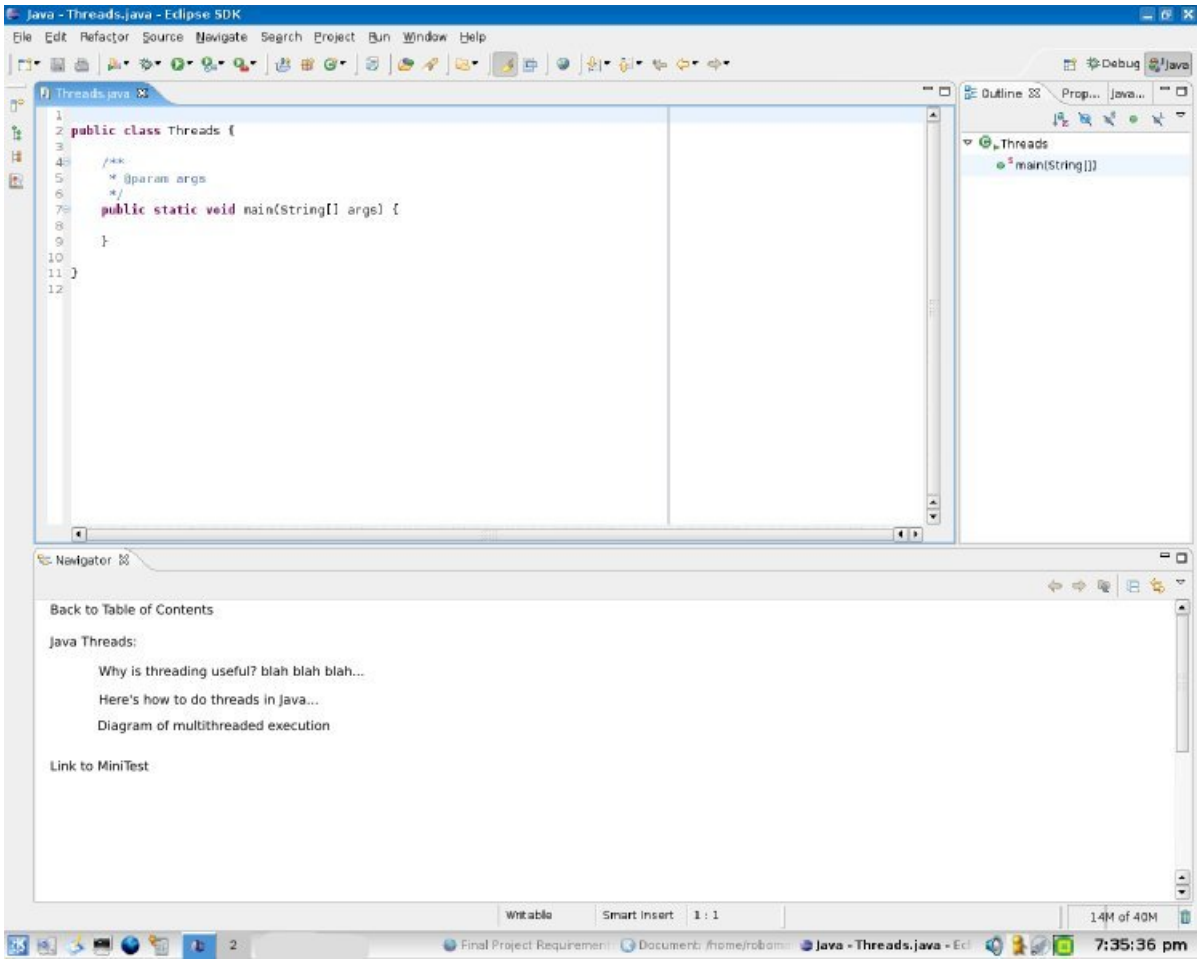

Figure 1: A screenshot of what the program will look like to the user. Note that the bottom panel is where tutorial text will be displayed.

the important concepts at hand. The first step was to read the programming assignment instructions from a webbrowser screen. The second was to do some programming and compile their results. This often required resolving syntax errors. If the student was confused about what to do, they changed to yet another window (this time a webbrowser as well) and read the online course notes. All of this navigational excise could be removed if the tutorial and assignment information was already contained in the same window which they user was programming in.

4 Results and Conclusions

Although we were unable to test our design implementation due to significant time constraints, we were able to draw some conclusions from previous research that we had done through interviews and observations of students actually learning C++. For most students, reading the text was not

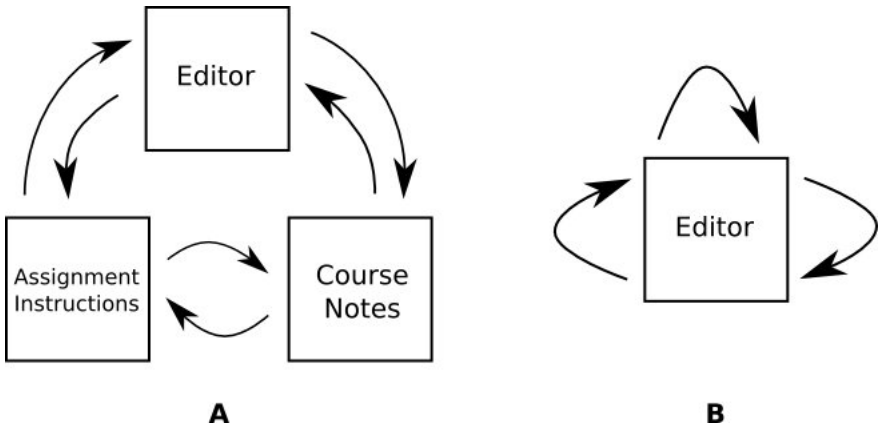

Figure 2: In the usual tutorial workflow (A), three windows and transitions between them are needed. In the Eclipse plugin tutorial (B), all three steps can be conducted without a window transition.

nearly as helpful as actually doing some coding to solidify concepts like syntax and object manipulation. Also, most students liked receiving immediate feedback about mistakes that they had made rather than having to read through compiler errors and guess at what they had done wrong. Our solution addresses both of these issues by encouraging the users to experiment with real code right inside the Eclipse environment and by utilizing the realtime syntax checking of the Eclipse environment to give the user instant feedback when they have made a mistake.

In addition to encouraging experimentation with real code, our solution also eliminates much of the excise found in the current workflow by putting all of the necessary elements in one window. This allows a user to concentrate more intensely on the material and concepts when their flow is never broken by unnecessary window switching. This also a major improvement over using a book, which poses an even greater amount of excise than switching between webbrowser windows.

We believe that this is a good first step in integrating the process of learning a new programming language with an advanced integrated development environment. The easy extensibility of Eclipse through plugins allowed us to experiment with a new pattern in learning programming language concepts that may prove to be a great step forward. We hope that this plugin may be used in a real student learning situation in the future. At such time, we may be able to judge the actual effectiveness of integrating learning materials with an integrated development environment.

References

[1] The official Java tutorial
`http://java.sun.com/docs/books/tutorial/`

[2] Don't Fear the OOP!
A tutorial which uses the metaphor of writing a novel
`http://sepwww.stanford.edu/sep/josman/oop/oop1.htm`