The Universal Computer: Introducing Computer Science with Multimedia

by

Glenn D. Blank, Robert F. Barnes and Edwin J. Kay

Table of Contents

Preface

This textbook with its accompanying multimedia has two major goals:

(1) To explore the **breadth of computing** as a discipline. A narrow concentration on programming can lead to a misconception that computer science is little more than programming. (The first chapter opens by addressing many common misconceptions.) There's more to astronomy than looking through telescopes: there's *theory*, such as relativity and cosmology, there's *experimental method*, and there's *design* of special-purpose instruments and probes. So there is more to computing than just programming: there's *theory*, such as the study of the limits of computation and the complexity of algorithms; there's the *experimental method* of programming and measuring the performance of programs; and there's the *design* of effective systems in software as well as hardware. Like older sciences and mathematics, computer science emphasizes *abstraction*, or pulling out the essentials. (The *Report of the ACM Task Force on the Core of Computer Science*[1] emphasizes abstraction, theory and design in the study of computing.) This book attempts to follow the recommendations of *Computing Curricula 1991* and *Computing Curricula 2001*[2] by introducing a breadth of knowledge areas of computing, ranging from computer architecture to artificial intelligence. Social, professional and ethical issues are explored in a chapter, as well as in exercises marked "*social/ethical*" throughout the book. Chapters and multimedia also both include many exercises, marked "*explore*", designed to get students to look beyond the book, by exploring the web or library databases. The breadth component of this book makes it suitable for many CS0 courses.

(2) To encourage students to think about and practice **software development as systematic problem solving**. We will make every effort to discourage "blind hacking"—that is, trial and error at a computer terminal. Good programming and problem solving in general requires planning and organization. Chapter 2 examines alternative approaches to problem solving, then gives students an opportunity to practice these approaches using a relatively simple, graphical robot simulation called "Knobby's World." Knobby is a cousin of Karel and Karel++, designed to introduce programming as problem solving. Chapter 4 studies the software development life cycle and a modern, object-oriented approach to problem solving and software development.

A *future* book will introduce programming in **Java "objects first"**. Java is a modern programming language widely used for developing large software systems as well as applets running on the web. By "objects first" we mean that the first things student learn about Java is how to manipulate classes and objects, rather the details of the language. Pedagogically, we believe that

[1]Denning, Peter, et al. *Report of the ACM Task Force on the Core of Computer Science*, ACM Press, New York, 1988. Also known as the "Denning Report." Reprinted in part in *Communications of the ACM* 32, 1 (January 1989) and in *Computer* (February 1989).

[2]Tucker, Allen B., et al., *Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force*, ACM Press, New York, 1991 took a strong position recommending a breadth-first approach to introducing computer science. *Computing Curricula 2001* (www.computer.org/education/cc2001/steelman/cc2001/chapter7.htm) acknowledges that "Even though the *Computing Curricula 1991* report argued strongly for a broader introduction to the discipline, the majority of institutions continue to focus on programming in their introductory sequence." Nevertheless, the new report notes that a "breadth-first" approach can "provide a more holistic view of the discipline, many computer science educators have argued for a "breadth-first" approach in which the first course considers a much broader range of topics." At Lehigh, we are able to cover both breadth and programming, with the help of multimedia.

students learn best what is taught first, and "object think" is what we think students ought to learn best, not "Hello World." Experimental evidence confirms that students can learn Java "objects first" with the BlueJ programming environment, especially with our multimedia.[3] We plan to deliver a future book on Java in 2004. (Interested parties may contact us about our manuscript and multimedia in progress.) The multimedia for the first Java chapter will be available with this book, as a way to give beginners a taste of object-oriented programming, after Knobby's World.

At Lehigh, the first author teaches two different first semester courses: 1) a CS0 course for non-majors, based on this book, and 2) a CS1 course for majors and minors, which introduces both the breadth of computer science using this book and Java programming using a different book. Breadth of CS and Java programming may seem like a lot to cover in one semester, but in our experience the multimedia makes it possible to learn enough about the breadth topics without devoting much lecture time to them. Interleaving breadth/problem solving and programming material gives students more time—breathing space, if you will—to learn Java programming by doing. Getting a computer to do what you want it do normally takes time: time for planning a solution, time for coding your solution, and yet more time for stamping out little "bugs"—the errors that lurk in code or even a conceptual solution. We strongly encourage students to get started on programming assignments early! The programming exercises integrated into the multimedia chapters will help students get started. While the Java programming chapters assume a particular order, instructors may cover the breadth chapters in just about any order.

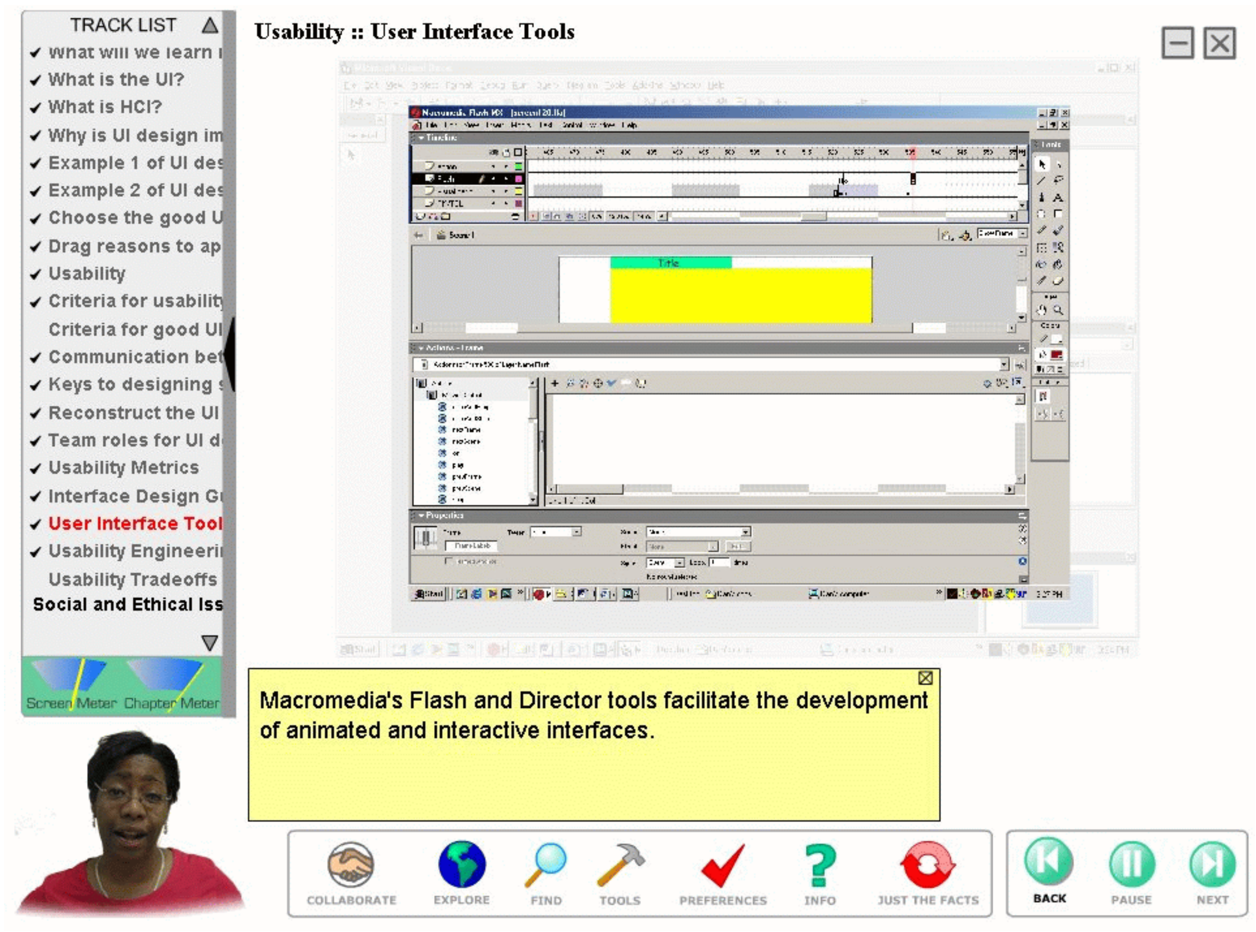
Multimedia

The multimedia which accompanies *The Universal Computer* is complete enough that students can learn from either the book or the multimedia, in either order, depending on their learning styles and/or instructor assignments. The multimedia content explains the salient material of each chapter, then reinforces concepts with interactive exercises, simulations, constructive exercises, and quizzes. Figure 1 shows the user interface and a sample screen. Here are some of the features of the user interface:

- The TRACK LIST on the left displays the content of a lesson as a sequence of screens. The menu uses check marks to show progress and highlights the current screen in red. At the bottom of the menu there are progress indicators through the current screen and chapter.
- Multimedia **personae** on the lower left model a diverse community of teachers and learners. The personae currently include three professors (one is shown) co-teaching the course, a teaching assistant, a reference librarian, and three students. In addition to graphical images, they speak in audio and/or text boxes. Personae model students and instructors studying material together, working through interactive exercises, and suggest exploratory research on relevant topics using online information.

---

[3] Experimental results in Glenn D.Blank, Willam M. Pottenger, Shreeram A. Sahasrabudhe, Shenzhi Li, Fang Wei and Henry Odi. Multimedia for Computer Science: from CS0/CS1 to Grades 7-12, submitted to *Ed-Media 2003* and available at http://www.cse.lehigh.edu/~cimel/papers/EdMedia03.doc. While developing *The Universal Machine* (the precursor of this book), the lead author taught CS1 to 70-80 computer science majors and non-majors. From the first year, for which only the manuscript of the book was available, to the second, in which an incomplete version of the multimedia was introduced, mean final examination scores improved about six points. Knowledge of breadth topics showed notable improvement. In the third year, with the complete first edition, mean final examination scores improved another seven points.



Macromedia's Flash and Director tools facilitate the development of animated and interactive interfaces.

Figure 1: Screen capture from *The Universal Computer*'s multimedia

- The COLLABORATE tools will facilitate network-based interaction with instructors, teaching assistants, librarians or other students. Chat, remote-controlled SHOW ME sessions and a multimedia FAQ of recorded SHOW ME sessions will encourage students to get help.
- The EXPLORE button facilitates inquiry-based learning, via directed queries on the web, adding to the *exploratory* exercises in the book. A state-of-the-art emerging-trends text mining and visualization tool will help students trace emerging trends as their interest and utility grow over time.
- The FIND button will bring up a search tool and a glossary of terms.
- The PREFERENCES icon presents a panel of options letting the user adapt the environment according to his or her personal learning style, including turning text boxes or audio on/off, toggling auto-advance or wait for next page, setting the timing rate where there is no audio narration, etc. A user may change these settings at any time during a session.
- A JUST THE FACTS mode lets users switch to viewing non-interactive content (text and graphics) presented in HTML pages. From there, one can switch back to rich media mode via hyperlinks anchored to the corresponding Flash page. There are also links to interactive screens, which remain in Flash. JUST THE FACTS mode, besides catering to some learning styles, requires less bandwidth, and may be useful for reviewing the material quickly.

The multimedia for *The Universal Computer* has been designed to accommodate diverse learning styles. Do play with the PREFERENCES options to suit *your* style! By giving students different ways to learn material, we hope to attract more novices, especially women and minorities, to computer science. It supplies sound and animation for sensory learners, while letting verbal learners disable sound or switch altogether to a "just the facts" mode. Interactive materials include learner-controlled simulations of algorithms, links to programs that students can try immediately after learning related concepts and before exercises that make sure the learner has studied the programs, constructive exercises in which students build programs or models by dragging pieces into place, and inquiry-based exercises in which students learn by doing research, using the web.

While our approach is to present enough didactic material in the multimedia that it can be a stand alone learning experience, interactivity is frequent and rich in *The Universal Computer*. Personae provide feedback to all responses.

Using the multimedia

The multimedia is deliverable either via CDROM., the internet (http://cimel.cse.lehigh.edu or www.cse.lehigh.edu/~cimel/prototype.html), or intranets set up at other institutions by arrangements with the authors. Since all content plays through a web browser (Microsoft Internet Explorer, plus a Macromedia Flash plug-in), it looks the same however it is delivered. An advantage of web-based delivery is that once users log in, their activities may be recorded in a tracking system; instructors and researchers will have access to tools visualizing a learner's activity, or reporting the result of quizzes. An advantage of CDROM-based delivery, on the other hand, is that it removes bandwidth constraints. (The multimedia, especially audio and video, can be sluggish playing through 56K modems; high speed connections or CDROM are strongly recommended.)

The CDROM disk and new web site make this a *multimedia* textbook. Beyond supplementing the textual material presented in this book, the disk includes more illustrations, animations, speech and other sounds, short movie clips, exercises and exercise solutions, bibliographic references to reading materials via the web or in a library, and tools that will help you learn to program in Java. Print text and multimedia can complement each other. The print text presents the core material of the course plus recommended exercises. We have not simply put the print text and source code onto a CDROM. The multimedia software emphasizes what is hard to show in the static medium of text: animating processes dynamically, illustrating abstract concepts concretely, solving problems interactively.

We have designed this book with a particular audience in mind: a class that mixes potential computer science majors with non-majors—because that's the kind of class we teach at Lehigh University. We want to encourage both those with those with little or no computing experience (even those who might be a little intimidated by computers), as well as those with more experience, to explore. We have installed the multimedia in a multimedia-capable computer laboratory which students attend once a week. We devote about one lecture per "breadth" topic and two or three per programming topic. We believe students can learn much of the "breadth" material and text without attempting to cover it all in detail in lecture. Multimedia thus allows instructors and students to get the big picture, rather than focusing too narrowly on programming.

Some schools may distinguish between majors and non-majors. A course that is more oriented toward computer science majors may want to put more emphasis on Java programming than we do and devote less class time to the "breadth" material, assigning multimedia chapters for students to study on their own or in labs. An instructor who feels it is urgent to get started with programming even earlier than we do could possibly skip programming in Knobby's World. Students can benefit from the illustration that this world supplies for non-programming concepts later in the book without necessarily having written Knobby's World programs. The multimedia systematically takes students through core Java concepts, then guides them through study of actual Java programs (by running them to see what they do, then answering the questions about what happened), then *hands-on* programming of related problems, using BlueJ or Javaedit. We believe multimedia learning is a major advance over getting source code on a diskette—and a lot more work to produce!

On the other hand, a course for non-majors might want to give less emphasis to programming and give more emphasis to the "breadth" material. In a CS0 course, Knobby's World may serve as a "gentle introduction to programming." The syllabus for Professor Blank's course is at www.cse.lehigh.edu/~glennb/csc10/syl.htm.

Exercises in the textbook can serve variously as homework assignments, in class demonstration or discussion items (we include many open-ended conceptual exercises for this purpose), and/or examination questions. Answers to many of the exercises, including all the review questions at the end of each chapter, are provided on both the CDROM and our Web site.

Acknowledgments

"Give thanks to the LORD, for He is good, and His love endures forever." Psalm 118:1

A new textbook is a considerable undertaking, but a multimedia textbook must be an order of magnitude more complex! We have learned much and benefitted much from working with a small army of editors, writers, fellow faculty members and students.

First, we repeat our thanks to those who helped with *The Universal Machine*: to Connie Coleman, our multimedia art director, to student multimedia developers John Chapman, Scott Fitch, James Nixon, Lingan Nguyen, Jason Stanford and Beau Sullivan, and to Knobby's World and LOOKOUT programmers Paul Martino, Venkatesh Rao, Louis Tanzos, and Jesse Thilo, and to Denise Gürer, then of Stanford Research Institute, for her extensive comments as well as her contributions to the "Women in Computing" theme in the multimedia. Thanks also to Betsy Jones at WCB/McGraw-Hill, who granted us permission to use the materials of *The Universal Machine* for our new book.

Second, we thank those who are helping with *The Universal Computer*: to the National Science Foundation (grant # EIA-0087977) and the Pennsylvania Infrastructure Technology Alliance (PITA), to student multimedia developers David Goldfeder, Martin Herr, Harriet Jaffe, Sumit Jain, Adam Kinnear, Andrew Mall, Sharmeen Mecklai, Jonathan Morgan, Soma Roy, Shreeram Sahasrabudhe, David Servas, Daniel Shire, Althea Smith, and Fang Wei.

Finally, we especially thank our wives, and our children, extended family and friends, for their love and patience while we buried our heads in this project!

G.D.B.
R.F.B.
E.J.K.