

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Chapter Eleven

Multimedia Authorware - Creating Learning Tools

The most important method of education always has consisted of that in which the pupils were urged to actual performance.

Thomas Alva Edison (1847-1931)

It's the software that's hard.

Michael Crichton (1942-)

The best way to predict the future is to invent it.

Alan Kay, father of the GUI

LEARNING OUTCOMES

Trained and experienced teachers would be the best people to create computer applications for use in schools. They have studied education theory and methodology. They have applied what they have learned in the classroom. They understand what will work and what will not when it comes to helping children learn.

However, most practicing teachers have neither the time nor the inclination to get involved in the development of applications software for computer-based learning. This is understandable enough. Planning, designing, and implementing such applications using programming languages such as Java or C++ demands an enormous amount of effort on the part of even technically-skilled and motivated teachers.

Fortunately, for the most part teachers do not need to create their own educational software. General purpose productivity and utility software for word processing, data management, communications, graphics, presentations and so forth are created by companies such as Microsoft, Apple, Adobe, and Riverdeep who employ whole armies of skilled programmers and education consultants. Much of their software has application in schools. Also, companies such as Sunburst, Scholastic, Tom Snyder and Brøderbund specialize in the development of software (courseware) for the K-12 education market.

But it is useful for pre-service or in-service teacher to know something of what goes into the planning, design, implementation, and support of software systems. Not only will it help them appreciate the effort involved, but it also will help if and when they either start to develop their own applications using authoring software, or need to advise students engaged in similar projects.

In recent years, relatively straightforward development tools called authoring systems have been introduced to simplify the task of constructing learning modules or systems. This chapter will examine some of the better known tools, though it is beyond the scope of this book to teach the

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

reader how to use them. Readers whose appetite is whetted by this introduction may want to work their way through the tutorials for *PowerPoint* which are part of the sets of tutorials for Microsoft *Office 2000*, *Office XP*, and *Office 2003*, an optional accompaniment to this text. The tutorials are available free of charge on-line at the following URLs:

Essential Microsoft Office 2000: Tutorials for Teachers

<http://www.pitt.edu/~edindex/Officeindex.html>

Essential Microsoft Office XP: Tutorials for Teachers

<http://www.pitt.edu/~edindex/OfficeindexXP.html>

Essential Microsoft Office 2003: Tutorials for Teachers

<http://www.pitt.edu/~edindex/Office2003frame.html>

Alternatively, the Microsoft makes available online or printed tutorials and demonstration disks. Here is an outline of the topics to be covered in this chapter.

- Introduction: programming is hard
- Why should teachers learn about software development?
 - Some historical background on programming languages
 - Knowledge is Power
- Characteristics of quality authorware
 - Quality Authorware puts the non-technical user's needs first
 - Quality Authorware is crafted with care and is relatively bug-free.
 - Quality Authorware makes the computer appear to be almost human in its responsiveness and anticipation of user needs
 - Quality educational software will eventually have to move to far more powerful computer systems than even those that are state of the art in classrooms today
- Authoring programs: How teachers can get involved
 - *Kid Pix Deluxe*
 - *HyperStudio*
 - *PowerPoint*
 - *TK3*
- Programming on the Web
 - HTML, JavaScript and JAVA
 - Web page Editors
 - Online Solutions, including blogs and wikis
- Authorware Essentials

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

INTRODUCTION: PROGRAMMING IS HARD

Teachers in schools k-12 are not expected to develop computer software from the ground up. After all, creating quality software is a tough task for a professional software engineer. It is accepted in the software development industry that no program is free of errors, or, to put it another way: "bug-free software is an oxymoron" (Meyer, 1992). Gleick (1992), while noting the importance of the software industry to the American economy, almost in the same breath reminds us that "bugs¹ are its special curse."

The fact is that programmers are human. The software systems they develop are recognized as the most complex that have ever been built (Olson, 1985). Even what appear to be simple programs such as word processors or spreadsheets may today have close to a million instructions. Software development is, quite simply, a mammoth undertaking.

A few teachers develop their own educational software from scratch. One such is Donna Mason, a computer education teacher and lab coordinator at the Alice Deal Junior High School in Washington, D.C. When she started teaching at Deal in 1983, there were no computers. When the school did acquire computers, there was no software to run on them, so she developed her own. The system she wrote, *MicroWorks*, was later published. In 1988, Donna was named a Christa McAuliffe Fellow, and in 1991, she was chosen by *Electronic Learning* magazine as one of "10 Who Made a Difference" (Electronic Learning, 1991).

But Donna Mason is the exception that proves the rule: teachers for the most part are not programmers. Nor should they be, since computer programming is a full time job of its own. Of course, some teachers who specialize in mathematics or computer studies are not only qualified, but they usually have a natural ability to write the step by step instructions that tell a computer what to do. A small percentage of students will also be enthusiastic about programming and might even create useful software applications.

WHY SHOULD TEACHERS LEARN ABOUT SOFTWARE DEVELOPMENT?

Our concern in this chapter, however, is with the majority of teachers. We need to examine to what extent all teachers can be expected to get involved with the development of software and other higher level learning systems such as multimedia applications.

Some Historical Background on Programming Languages

Software development occurs at several levels. Traditionally, programmers talk about low level programming, which uses programming languages (machine language and assembly language) which are close to, or at, the internal machine level of the computer. Low-level programming involves the most detailed and extensive sets of instructions to tell the computer what to do. Such programming is of no concern to the vast majority of teachers.

¹ A "bug" is a small flaw in either code or the structure of a computer program. The word "bug" has been used to describe errors in computer programs ever since Grace Murray Hooper, while working at Harvard on software for one of the early electronic computers, discovered that the problem with her program was caused by a moth that had gotten lodged in one of the switches or relays inside the machine, preventing the switch from flip-flopping on and off. Once she removed the bug, the computer program ran just fine.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

At a higher level, programming involves the use of languages such as Basic, Pascal, C++, Java or Logo--to name only those most commonly learned in k-12 schools. These are examples of what are known as high level languages (HLLs). They still require some understanding of how the computer works, and they also involve large numbers of instructions to get the computer to do anything particularly worthwhile. But they are closer in syntax and "vocabulary" to human languages.

At another level are object-oriented programming (OOP) languages like RealBasic and VisualBasic. These eliminate some of the difficulties with program structure and "end result interface" by allowing the programmer to use graphical assistants, code snippets and templates; the program is built visually rather than entirely through code.

Probably fewer than about 1% of teachers have ever developed a software application that has more than a few hundred instructions. But many teachers over the last decade, in their efforts to learn about computers, have toyed with programming at this level as part of computer literacy courses. Perhaps you are one of these teachers, and perhaps your experience has taught you that you are better off leaving this kind of programming to others!

At a still higher level, there are languages such as SQL (Structured Query Language)² which were specially designed to simplify access to data in large database management systems (DBMS), and ProLog, which is highly mathematical in structure. These languages are known as very high level languages (VHLLs). Since these languages and their applications are predominantly of use to researchers, corporations and central educational administrations, it is unlikely that teachers would need to be familiar with them either.

Early multimedia authorware was a bridge between high level languages and today's multimedia authoring tools. Apple's *HyperCard* and early versions of LCSl's *MicroWorlds* and Solutions Etcetera's *SuperCard*, for example, allowed users to create stand-alone applications that included scripted media elements, such as animations, graphics, sounds, and hyperlinks. It was still necessary, however, for users to learn a "language." These applications have not disappeared; in fact, they have become more refined, enabling users to create complex learning environments. The newest generation includes Alan Kay's *Squeak*, *MicroWorlds EX* and *MicroWorlds JR*, and Macromedia's *Flash*, multimedia problem-solving and programming exploration environments that can be used by students as young as pre-K. These applications are providing web-based collaboration tools for the constructivist learning environment. They have the advantage of being powerful learning tools, but the disadvantage of themselves having a somewhat steep learning curve. They are not tools for casual or "sometimes" use in the classroom.

Luckily, there are applications development tools, such as the authoring tools that we will profile later in this chapter, which require next to no understanding of how the technology goes about its work. Users work from "buttons" and menus and take advantage of the "drag and drop" capabilities of the newest operating systems. The programming happens entirely in the background, thus releasing the "programmer" to concentrate on the application, educational or otherwise, for which the courseware is required.

² You may have run across MySQL, which is a database application used to create web-based forms. In a user-friendly format, it is often provided as a service to clients by webhosting companies.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Examples of these authoring tools are, Broderbund's *Kid Pix Deluxe*, Sunburst's³ *HyperStudio*, Microsoft's *PowerPoint*, and Night Kitchen's *TK3*. We will briefly examine each of these applications, along with some authoring tools for the Web.

Teachers and students should learn to use one or more of these tools, or similar tools, so that they can use them for teaching and learning. When the teachers, through their own experience, know the potential of an authoring tool, they can more effectively direct students in designing worthwhile projects that take advantage of the tool's full range of capabilities. Authoring tools provide a discovery-based or constructivist environment for students of all ages to conduct research, gather data, organize their thoughts, think creatively, and share their newfound knowledge with others.

Knowledge is Power

Because of this variety of approaches to programming, different teachers will have different reasons for wanting to know how to go about developing software applications. Here is a cross-section of those reasons. It will be interesting for you, the reader, to see where you fit in this spectrum of rationales.

- Programming a computer helps one understand how the computer works, which leads to a greater sense of control when using the computer for more general and pre-packaged applications.
- Knowing how to program a computer helps one appreciate the work that goes into the systems that are used by teachers and students in educational settings.
- If one knows how to use software development programs such as authoring systems, one can develop applications that are tailored to one's specific local needs.
- Students like to develop their own programs/presentations using multimedia tools. If teachers understand how these tools work, they can be a better resource for their students in the classroom.
- Teachers who understand what constitutes a well developed program will be better at selecting applications that will be effective educational tools.
- Teachers who have survived what might be the trauma of trying to figure out the correct logical sequence of instructions that constitutes a computer program will be sympathetic towards students when they have difficulties with the same process--or any learning process, for that matter.

CHARACTERISTICS OF QUALITY AUTHORWARE

Quality authorware puts the non-technical user's needs first

³ Kid Pix was originally created by Broderbund, which is now a sub-division of Riverdeep. HyperStudio was originally created by Roger Wagner, but has since been sold several times. This is not unusual in today's software marketplace.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Usability engineering is the application of human factors (also called "ergonomics") to system design. The most time-consuming aspects of software development often have to do with how well the product will fit the "event world" for which it is created (Debons, 1988). The "event world" of k-12 schools is populated by end-users who do not expect to have to learn how a machine works in order to put it to work.

Key to lay acceptance of computer technology are ergonomic considerations that dictate that the designer of a system understand, and cater for, the cognitive and physical constraints that the human user brings to machines. What Tufte (1990) says about the graphical visualization of information applies equally well to software in general and to educational software in particular. "Clutter and confusion," he says, "are failures of design." The best software applies principles of ergonomics to all aspects of design so that the end product--the program--is as easy as possible to use.

Bailey (1989) notes that "a good computer-based system does not require extensive assistance from people." It should be designed in such a way that it marries into the paradigm of normal, everyday practice: business, educational, or otherwise. This emphasis on usability is a relatively recent phenomenon. Systems in the past that had an unreasonable learning curve associated with their use were successful only because there was little alternative for people who wanted to use the computer to increase productivity and control. Today, however, it is understood that the benefits of advances in technology should be made available to all, not just the privileged few who were able to master the intricacies of traditional systems.

Moreover, as Stahl (1986) explained in his article enunciating the principle of "least astonishment" in interface design, "system performance is directly dependent on user performance." When users are comfortable working with a system, they will be more productive. Some orientation is, of course, inevitable and time must be allocated for this in the introduction of a new system. But as Arthur Young & Co. discovered when they selected Apple's Macintosh as the standard system for their accountants, the adoption of a system that weds functionality to ease of use can bring a convincing bottom line benefit to the organization in terms of reduced end-user support costs and increased end-user productivity (Garfinkel, 1987).

Fortunately, the last few years have seen the industry-wide adoption of standards for the software interface which have greatly simplified interaction between the machine and the people using the machine. We discussed these graphical user interfaces (GUIs) in chapter 3. They require the user to work in the context of *windows* in which they use the mouse to *point* and click on *icons* (pictures that represent processes and files) and select from *menus* or lists of things to do. The **Windows, Icons, Menus and Pointers** that are the components of the GUI give us the acronym **WIMP**, which captures the essence of modern easy-to-use computing environments. Just about all the information needed to know about how to use an application is there in front of the user on the screen. Once one application has been learned, the learning curve to master other applications is considerably less steep because the interface is familiar across applications.⁴

⁴ Assistive technologies and applications that take advantage of them are perhaps the most profound present-day illustration of good design. Applications such as Cricketsoft's *Clicker* and Intellitool's *Classroom Suite* meld multimedia and design to make reading and writing assessable to all elementary level students.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Teachers want to be able to focus on the pedagogical aspects of courseware, not the technological ones. Teachers do not want to have to learn technical material in order to use software, nor do they want to have to teach that technical material to their students. One way to help technically-shy teachers get involved with technology is to introduce them first to systems that are both easy to use and at the same time manifestly useful for some teaching or learning activity.

Quality authorware is crafted with care and is relatively bug-free

It is almost impossible to guarantee that the software one acquires will work as it should all the time. This is because it usually takes hours, days, weeks, even months of use before many bugs show up. In fact, a corollary of Murphy's Law says that bugs show up when least expected and at the worst possible time. For example, you have just about finished typing up a ten page term paper in the computer that you have not yet thought to save. The system freezes up. Yikes! Now what? You may never know if it was the applications software you were using that caused the problem, or some utility program running in the background. Who cares? All your work is gone!

It pays to know the reputation of the company that has developed software, and the only way to keep track of this is to read software reviews that appear in educational journals and to check with colleagues in your own or other schools who have used the software. You may also be able to run your own test by downloading a free demo copy of an application from the producer's website. Such full-functioning trail versions are increasingly available. Another thing to look for is a "beta" version of a new application. These are not guaranteed to be "bug free," but they have passed the first few rounds of serious "alpha" testing. Betas are always free and often very usable.

If you ever develop your own software, or if you are working with students who are developing projects around multimedia systems, the most important recommendation is to pay attention to detail. This attention to detail is the hallmark of the best professionals in all walks of life. The second most important recommendation is to maintain your focus upon an educational outcome. Remember that computer applications are excellent because they are part of an excellent lesson, unit or curriculum. These points together mean that developing software and applications is hard work!

Quality authorware makes the computer appear to be almost human in its responsiveness and anticipation of user needs.

New artificial intelligence (AI) and Fourth Generation languages⁵ such as those used to create operating systems and authoring software, and better yet, natural language itself, greatly simplify human interaction with the machine. *AppleWorks*, for example, along with other integrated software or suites of software such as *Microsoft Office*, are designed in such a way that they know whether a user is involved in word processing, or database management, or another of the productivity functions that come with the package. When the user wants to create a diagram, or import some artwork, the user simply starts working with the graphics tools, and the system handles the switch to that module of the software.

⁵ A Fourth Generation programming language is a high level language that uses "natural" expressions, such as *Process* and *Print*, to encapsulate many lines of code.

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

As we will discuss later in this chapter, authoring software make software development relatively intuitive, allowing the developer to focus on the purpose for which the application is intended rather than on the process of development itself.

Quality authorware will eventually have to move to far more powerful computer systems than even those that are state of the art in classrooms today.

The paradox is that, in order to optimize the human resource by designing thoroughly ergonomic systems, we must rethink the design of traditional computer architectures so as to improve the performance of the machines. Execution of one instruction at a time, for example, is too slow for the kinds of applications envisaged by researchers in AI.

The processing necessary to allow for the simplest of natural interactions stretches the capabilities of many currently available computing machines. Graphical interfaces have been demonstrated to be among the most natural for the human user, but such interfaces are extremely demanding of machine time and space. Natural language interfaces, too, which allow users to interact with the machine in their own native tongue, apart from presenting enormously intractable linguistic difficulties, are once again very demanding on machine time and space.

The most powerful computer in the world in 1946--the ENIAC⁶--was equivalent to the pocket calculator built in the 1970s. Today's most powerful computers, capable of processing trillions of instructions per second (teraflops⁷), will inexorably become the personal computer of the not-too-distant future. Already the PC is capable of executing billions of instructions per second (gigaflops) and it is just a matter of time before the teraflop barrier is crossed, leading to a quantum leap forward in terms of PC functionality. Moreover, the ongoing increase in processing speed is accompanied by an ongoing increase in portability. Not only will the classrooms of the future have very fast computers; they will have very small and entirely portable computers.

With such machines in place we will see a world open up which will make the advances made to-date in computer technology seem puny by comparison. Most important of all, these machines will enable the creation of software systems that will help educators realize the full benefits of technology in the classroom, without the kind of mental trauma which has given so many teachers pause up till now. But Michael Crichton's caveat, quoted at the outset of this chapter, will continue to hold true: "It's the software that's hard".

AUTHORING PROGRAMS: HOW TEACHERS CAN GET INVOLVED

Chapters 5 through 10 examined in detail what teachers need to know about computer use in the classroom. In summary, teachers are proficient using technology when they feel comfortable about:

⁶ The electronic numerical integrator and computer (ENIAC) built by Presper Eckert and John Mauchly, and profiled in Appendix A (The History of Computers).

⁷ "Tera-" is the prefix that stands for ten to the twelfth power, or a trillion, just as "giga-" is the prefix for ten to the ninth power, or a billion. So a terabyte is a trillion bytes. "flops" is an acronym for "floating point operations per second." Thus "teraflops" is the word used to describe a trillion instructions per second.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

- working in a computerized environment;
- using computer-based tools to manage their professional activities in and out of the classroom;
- incorporating a wide range of computer-based learning into their curricula;
- using computer technology to establish close contact with parents;
- encouraging students to use communications technology to establish cross-cultural links with children and adults around the world;
- incorporating multimedia systems into the learning process.

These technology-proficient teachers will provide their students with an educational environment in which each student's individual and infinitely-varied talents can grow. If, on top of all this, these teachers are able to use, and direct student use of, authoring tools to fashion learning systems that incorporate hypermedia technology, they will have made the transition into the world of *Education For an Information Age*. This is not as difficult as it sounds. Let us look at some examples of authoring tools.

Traditionally, as explained above, programs are created by programmers. These are usually people with strong technical skills who have undergone extensive training. Authorware are programs which help people who are non-programmers develop applications for the computer. They are a natural outcome of the recognition that the best people to develop software are those who are experts in the area of activity or expertise for which the software is intended. This, indeed, was Bill Atkinson's vision when he developed *HyperCard*, the first authoring system for microcomputers and the prototype for most of those that have been developed since 1986. As Atkinson put it: *HyperCard* is "programming for 'the rest of us'." (Goodman, 1987)

People who understand business are the best people to write applications for business. Likewise teachers have considerable expertise and experience related to teaching and learning methodologies. They are, therefore, the best people to be involved in the development of all types of CAI. Indeed, many of the companies that currently develop software for schools were founded by teachers with the technical skills needed to design and develop software. Other companies that specialize in software applications for schools have teachers on their payroll either as full-time personnel or as consultants.⁸

Of the many examples of authorware available today, we will profile just four in the sections that follow. First we will profile *Kid Pix Deluxe*, the simplest and easiest to learn and, for these reasons, perhaps the most useful with younger children from a practical point of view. Then we will briefly examine *HyperStudio*, *PowerPoint* and a new application called *TK3*. These authoring tools are designed for both Apple and Windows systems, and are possibly the most versatile of the authoring genre of educational software. We will not be looking closely at *MicroWorlds* because of its programming component, but we urge interested teachers to undertake an investigation of Logo and this powerful learning environment.

⁸ This is also true for Internet based educational applications.

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Riverdeep|Broderbund's Kid Pix Deluxe

Kid Pix is an interactive drawing. Unlike COREL's *CORELDraw* or Adobe *Photoshop*, it is not a highly sophisticated drawing package. But, like Apple's *AppleWorks Paint* module, it is highly effective environment for creating lively, inventive graphics. It has the added advantages of a simple toolkit layout and a set of tools consistent with *AppleWorks*, Microsoft *Paint* and other basic paint applications⁹. *Kid Pix Deluxe*¹⁰ is a presentation package which includes a new version of *Kid Pix* as well as new tools to take advantage of processor speeds, system capabilities, and larger hard drives. It is useful for creating sound-enhanced slide shows of *Kid Pix* graphics, but it also imports still and movie graphics drawn from an infinite variety of other sources, including any movie that has ever been made, provided it is accessible in a digital format (and cleared for copyright).

As made clear elsewhere in this text, it is not the intention of the authors to advertise any particular product. Consider this a disclaimer to that effect! But *Kid Pix Deluxe* speaks for itself as a remarkably simple, yet powerful, example of authoring software. It should be made available as a learning tool in every elementary school classroom and computer lab. It helps if the classroom has at least five or six computers with CD-ROM/DVDs, so that the children can work in collaborative groups on projects drawn from the curriculum.

Storyboarding is a technique developed in the movie industry. Each storyboard is "a panel or series of panels with sketches depicting changes of action and scene" (Random House, 1991). Adapted for an interactive medium such as the computer, the storyboard can come alive with animations, video clips, and sound. The computerized storyboard can also interact with the user, prompting for responses to situations presented in the storyboard's action.

In *Kid Pix Deluxe* the user can work on any particular project with up to 99 storyboards, each of which is represented by a moving van which can be programmed to "carry" still or moving graphics, sound, text, and transition effects. Buttons on each slide in the storyboard provide access to each of these options (Fig. 11.1 next page). As you examine the illustration, play the role of a pre- or early reading student. After a short lesson, would you be able to make a short presentation?

- Click on the first button associated with a particular storyboard, and the user is prompted to select a still or moving image to be included with that storyboard. The graphics can be selected from commercial clip art saved as graphic files, *QuickTime*TM movies, or art prepared using a paint program or *Kid Pix* itself (this is where a teacher must be part of the process).
- Click on the second button, and the user can paste in prerecorded sound clips, such as a Martin Luther King speech, or use the microphone to record commentary, either on the fly or, ideally, from a prepared or "live" script.

⁹ *IrfanView*, a free paint program for Windows, uses a similar toolbar. This cross-over of tools is an important element of software usability.

¹⁰ *Kid Pix Studio* is an earlier, and very similar, version - it may still be available on many school computers.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

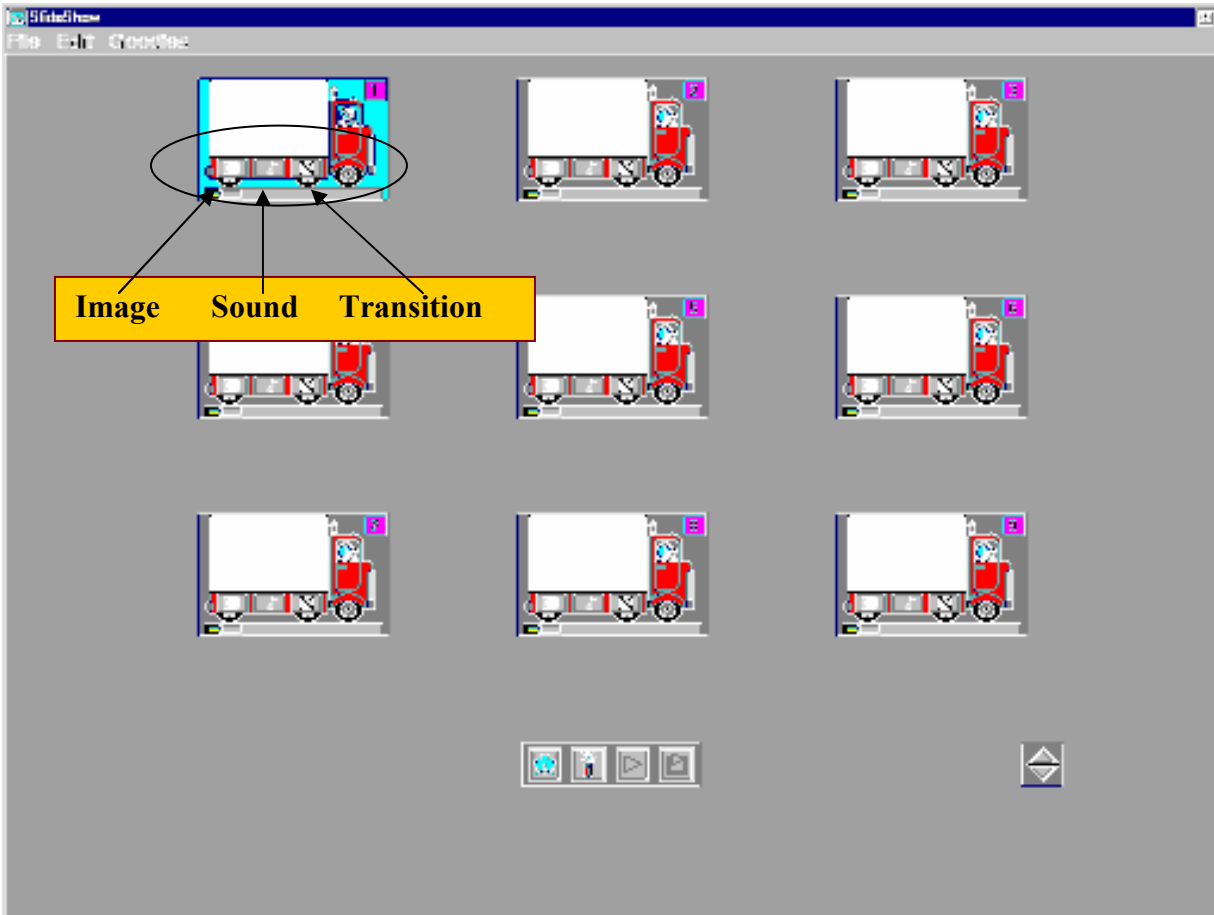


Fig. 11.1 *Kid Pix Deluxe* interactive storyboard showing the function of the buttons

- Click on the third and last button, and the user can select from a set of transition effects which will provide a smooth transition from the current slide to the next. Effects can be previewed.

A resourceful teacher will quickly learn how to use *Kid Pix Deluxe* to prepare learning materials on any subject matter for use in the curriculum. Still more resourceful teachers will show their students how to use this tool so that the students can use it to put together presentations which are the outcomes of individual or group project work.

Sunburst Technology's HyperStudio

Roger Wagner Studio's *HyperStudio* was a ground-breaking k-12 application designed with a familiar index card model: "Stacks" contain sets of electronic cards, a format that has not changed in the subsequent versions of the application. Each card can be programmed with buttons, graphics, sounds, text, and other multimedia elements, including media links to the Internet. Many media elements can be created with the application itself, which contains an animator, a sound recorder and a paint program. Many other forms of media can be imported into a card's environment. *HyperStudio* projects can comprise a single stack, or can be made up of

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

two or more stacks linked together. Like *Kid Pix Deluxe*, *HyperStudio* has built-in paint tools for drawing and artwork; like *Kid Pix Deluxe's* storyboards, *HyperStudio's* storyboard provides an overview of the project and links to individual cards, that in turn can hold links to graphics, sound, and so forth. But here the similarity ends.

HyperStudio is a more comprehensive application development package. It allows the user considerable flexibility in the design and content of stacks, limited, for all practical purposes only by the amount of primary and secondary memory available¹¹, the speed of the computer's processor, and the developer's time and creative skill. For this reason, it is more difficult to learn and use than *Kid Pix Deluxe*, largely because there are many more features to become familiar with. But, once learned, it is possible to create rich multimedia learning environments with application at all levels of education, from pre-K through college.

Logo provides a useful extension to *HyperStudio*. Any card in a stack can be linked to a Logo program. Since Logo effectively can be used to create any application, a *HyperStudio* stack can contain, quite literally, an infinite range of problem-solving functionality. New versions of *HyperStudio* enable the user to script directly in HyperLogo, a language very similar to Logo in syntax and vocabulary.

Another advantage to *HyperStudio* is its support of web-based projects. It is possible to Export stacks as .html and to upload them to web pages, playable on any computer platform with the free HyperStudio Player.

Microsoft's PowerPoint

PowerPoint is understandably the most used presentation program. It is part of the Microsoft *Office* suite of productivity software and therefore is available on well over 90% of computers. It is also the easiest presentation program to learn for developing effective presentations, learning modules, and other kiosk-style applications where the goal is to present information to an audience.

PowerPoint has many powerful features, but for education perhaps the most powerful features relate to its interactivity. As part of the *Office* suite, *PowerPoint* contains all of the editing, commenting and notation features built into Microsoft *Word* and *Excel*. Students and teachers can not only author presentations, they can collaborate and peer-edit. This makes it an important model for 21st century authorware.

It is strongly recommended that pre-service and in-service teachers be given the opportunity to learn to use this software since there is a very high likelihood that they will be able to incorporate it into their work with students k-12. Soon, every student will have a personal computer with *PowerPoint* on board and, as pointed out in chapter 6, *PowerPoint* can be a powerful tool for project-based learning of various kinds.

¹¹ If your computers are on a network, you must also consider network bandwidth and the size and configuration of the system storage device. Many HyperStudio projects have been lost because of network weaknesses.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Night Kitchen's TK3

New applications are being developed to take advantage of new Windows and Apple operating systems, which allow for a higher level of media integration "on the fly" and which require computers with larger hard drives and faster processors. *TK3* is one such application. Like the previous examples, it is cross-platform and able to import media files in multiple formats. Unlike the other applications highlighted, however, it does not create a hyperlinked "slide show." This application creates an e-Book, a self-contained, interactive multimedia text.

You will remember from previous chapters that there is a trend in Internet and server-based educational software toward the development of "media objects" that can be recycled and freely shared. *TK3* makes use of imported media resources, sound, video, image, font and text files, to build a compact "book" upon a template designed by the author. In addition, the application supports internal and external hyperlinking, file merging, and export to a compressed format.

We highlight *TK3* because it is a forward-looking application, tailored to teachers who are comfortable with the media tools highlighted in the previous chapter and eager to develop and distribute their own courseware in a self-contained mode (as opposed to an e-Learning mode).

PROGRAMMING ON THE WEB

We learned in Chapter 7 that the World Wide Web was invented in 1991 by Tim Berners-Lee. He created a language and other standards that enabled anyone to create a presence on the Internet that would be accessible from anywhere in the world. This included corporations (.coms), educational institutions (.edu), non-profit organizations (.orgs), the government (.gov), and everyday individuals¹². Tim Berners-Lee also set up an organization called the World Wide Web Consortium (<http://www.w3.org>) which manages the Web, developing new as well as updated standards to help everyone on the Web to get along.

The upshot of Tim Berners-Lee's work is that everyone who so chooses can share themselves with whoever is interested in what they have to share. This is both a scary and an exciting thought. As we noted in Chapter 8, there are lots of odd folks out there—people who have mind sets that may not agree in any way, shape, or form with ours! Of course, to these "odd folks" we may be odd, too. But that's another story... Fact is, everyone has a voice, and Tim Berners-Lee has given all of us a global audience. Good teachers take advantage of this beautifully simple, yet powerful, reality.

The World Wide Web is the single most powerful tool at every teacher's fingertips. The Web puts technology-using teachers in "24-7"—twenty-four hours a day, seven days a week—touch with each (yes, each) of their students *and* those students' families! This is an awesome fact, and one that might take some time to filter through, and alter, the mindset of the teaching profession as a whole.

¹² Currently, the list of possible computer domains that can be purchased in the US is: .com, .org, .net, .info, .biz, and .us. The .gov domain is reserved for state and federal government, and .edu is generally available only to universities and large non-profits that host their own sites.

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

Think about it... You are a teacher. You have a class of students of whatever age, and you want them to learn something dictated by the curriculum (which hopefully reflects state or national standards). You know the kids love working with computers, so you create your own personal website¹³ and use it as a supplement to what you do with the students in class. You include your own content that will guide your students to the information they must learn in order to attain the academic standards laid down. You also use your website to let your students' parents know what is going on in the class. You spend whatever time it takes to research the web and locate sites that have the content that ties in with the curriculum. If you are unable to find relevant content, you shrug your shoulders and create it yourself!

Whoa! Create it myself? You mean teachers should create web pages which contain the information content that they want their students to learn? Absolutely! Because once you, the skilled 4th grade, or 8th grade, or 12th grade teacher, have created such content and posted it on the web, it is immediately available to other teachers, near to home or around the globe, who can use it with their students, too.

So you get to share your excellence with the world. Think of the many excellent teachers who touch only the students that have a physical presence in their classroom. Think back to your own days in k-12 school. Remember the teachers who touched your life in beautiful ways. You know who they are. Now imagine how they could expand their influence over children and their parents if they use the web to reach that wider audience.

Technology in general and the web in particular thus help teachers duplicate excellence. The Internet gives new meaning to the old adage that teachers touch the future. Well-organized web-using teachers—and there are already many of them out there¹⁴—are pebbles into the pond of education, which create ripples that reverberate around the world.

HTML, JavaScript, DHTML, and JAVA

HTML (Hypertext Markup Language) was developed to standardize all aspects of content creation on the Web. Generally, pages displayed on computers that access the Web have been defined using HTML. HTML is a simplified language, not nearly as difficult to learn as typical programming languages such as C++ or even BASIC. This is because HTML gives the programmer much less direct control over the data that is being processed (distributed) on the Web. In the truest sense, it is not a programming language at all; it is a set of instructions that tells a browser how to display digital content. However, HTML still requires of the program writer a certain rigidity and care for detail that stretches the patience and skills of most folks. For this reason it is unrealistic to expect that more than a tiny minority of teachers will want to develop learning materials using "hard coded"¹⁵ HTML.

¹³ A "website" is a collection of web pages, hosted on a web server. Think of this as a set of documents inside of a folder, which it is, virtually.

¹⁴ Here is just one example: Tom Daccord's *Best of History Web Sites* began in the classroom and has become one of the best links for history teacher's in English-speaking countries - <http://www.besthistorysites.net/>

¹⁵ "hard coding" is a term that indicates the programmer is writing the code, in this case HTML, from scratch. As you will see in the next section, it is possible to generate HTML without hard coding a web page.

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

JavaScript is a simplified programming language used mainly to provide simple interactivity and "special effects" to web pages. Although it is possible to build scripts¹⁶ from scratch, most webmasters will begin with scripts made freely available online and adapt them to meet specific content and design needs.

Similarly to JavaScript, but more adaptive and therefore more powerful, is DHTML, a language that also created small scripts that are embedded in the HTML of a web page. Both languages work well with today's browsers, although JavaScript support is uneven. One important script source is Simply the Best Scripts (<http://simplythebest.net/info/dhtmlinfo.html>). With a little initiative, a good plan, and attention to detail, almost anyone can include JavaScripts or DHTML scripts in a web page.

JAVA is another kettle of fish entirely, a sophisticated programming language with much the same capabilities as C++—a full-blown computer programming language. You have probably run across JAVA in terms of "applets" loaded on certain web pages – if you look at the Status Bar at the bottom of the browser window, you can "see" them being loaded. These applets are used to create complex animation and interactivity. However, entire websites and stand-alone applications are also developed in JAVA, which is now the programming language assessed by the Computer AP test.

Why then even talk about HTML, JavaScript, DHTML or JAVA¹⁷ in the context of teacher-created applications for learning? Apart from anything else, it is good to know more than you need to know in order to do what you need to do. An awareness of what some of the more commonly used HTML "tags" ("instructions" in other programming languages) can come in handy when you want to tweak the contents of a web page or if you want to see how an attractive feature of someone else's web page was put together.

You can view much of the HTML and JavaScript source code for a web page in your browser by choosing the *View* menu option to view the *Source*, which is often the .html code that created the page you see¹⁸. This code is, of course, copyrighted by its creator (unless you read otherwise), much as a novel is copyrighted. However, it is OK to copy it for the purpose of learning. If you can understand what the HTML tags are doing, then you can play with them, or tweak them, in order to create the look and feel that you want for your page. In this sense a little knowledge can go a long way in terms of web page design. Of course, a little knowledge can also be a dangerous thing! You don't want to mess with HTML unless you know what you're doing¹⁹.

Think of it in terms of owning a car. You don't *need* to know anything about maintaining the car. You can just take it to a garage whenever you need anything done. It'll cost you more, but you

¹⁶ A "script" is a miniprogram, sometimes just a short instruction, that sits inside of the HTML on a web page.

¹⁷ These are by no means the only languages and protocols used to create web pages and web page interactivity. Some others which you may meet are: XML, Perl, cgi, pHP, and CSS.

¹⁸ There are many exceptions to this rule, the most common being pages viewed within frames and pages created with *Flash* or *Shockwave*, which are really animation objects instead of coded pages.

¹⁹ Changes that you make to copied Source code will not, of course, affect the original page. You can only alter a page on a web server to which you have direct access. One easy way to explore HTML is to create a *blog* or web page using an online host and "tweak" the template you select. We cover blogging in a later section.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

won't get your hands dirty. A few folks won't even pump their own gas and instead pay the price of full service. But if you know how to fill the gas tank yourself, and if you know how to change the washer fluid or the wiper blades, or if you know how to jump start your engine when the battery gets drained because you left the lights on... You get the point. A little knowledge can save you a lot of time and money.

It is beyond the scope of this book to go into the details of HTML or other programming languages, but there are many excellent tutorials on-line. Among the best can be found at the following URLs: <http://www.w3schools.com/html/default.asp> (for learning HTML), <http://www.scriptsearch.com/cgi-bin/jump.cgi?ID=3878> (a straightforward introduction to JavaScript) and <http://www.w3schools.com/dhtml/default.asp> (an introduction to DHTML). Teachers who are interested in blogging and wiki's should also consider pursuing CSS. A tutorial can be found at http://www.w3schools.com/css/css_intro.asp.

Web page Editors

Web page editors make it almost entirely unnecessary for you to know HTML. The best known web page editors today are Microsoft's *FrontPage* and Netscape's *Composer*. Another popular editor is Macromedia's *Dreamweaver*. These tools are designed to help you create web pages with a friendly, WYSIWYG (what you see is what you get) interface. They are like word processors for the Web. As such, they greatly simplify the whole process of working with hypertext (text that is "hotlinked" to other related information on the Web), media and hypermedia (text, images, sound, video that is hotlinked in the same way) in the non-linear environment of the Web. New tools, such as Apple's *iWeb*, further streamline the process, by providing templates and automatic uploading to compatible web spaces such as Apple's .Mac hosting service.

The web page editors use tool bars and simple point and click steps to create what would otherwise be very tricky objects, such as forms, navigation bars, tables or image maps²⁰, let alone hyperlinks. The editors are not difficult to learn how to use. What is difficult and very time-consuming is the whole process of creating well designed, well thought out, and well-constructed content.

Once again, it is beyond the scope of this book to go into the details of any particular web page editor. In today's teacher training or inservice workshop schedule, the use of web page editors should be part of the curriculum for all preservice and inservice teachers. The web is becoming an important learning environment, and who better than teachers to create content that is pedagogically sound.

Online Solutions

Consistent with the trend to move more and more of educational technology onto the web, web-based options exist for teachers and students wishing to create basic courseware. We looked at several of these in Chapter 8, in the context of e-Learning. With a low-cost, or even free, subscription, teachers can create basic hypermedia websites simply by using a browser. Some

²⁰ Image maps are graphics to which invisible hyperlinks have been applied.

EDUCATION FOR AN INFORMATION AGE
Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

schools are moving toward a CMS (Content Management System) that enables teachers to quickly and easily add content to a template provided for them by the IT services.

Additionally, many, if not most, of the multimedia productivity applications can Export or Save to .html, creating an "out of the box" web page or website. *Word*, *PowerPoint*, and *Inspiration* are especially adept at this task, which can be "buggy," but which is also an enormous timesaver, especially when teacher's can *upload* web pages to a school server quickly and easily. This is another instance when knowledge of HTML and website organization can be power.

Teachers wishing to explore one of the cutting edge solutions should investigate *blogging*, or weblogging. Originally designed as text-based online journals, blogs have now become multimedia in content. Free hosting services and applications such as Apple's *iWeb* make it possible for teachers to design entire units and courses around a blog, which can contain images and video, recorded narration or other audio, hyperlinks, and collaboration tools. Content, including voice recordings and images captured on a multimedia cell phone, can be seamlessly uploaded to a blog. Although currently in the realm of "tomorrow's classroom technology," this authorware solution is worth more than a passing glance. *Blogger.com*, *Blogdrive.com*, and *Moveabletype.org* are examples of free blog tools available to teachers and schools.

An even newer tool is the WIKI, a web-based learning environment in which every teacher and student, every visitor, can be an author. Teacher's can now create classroom wiki's using free online tools such as *Wikispaces.com*²¹. As the wiki protocol and underlying programming environment improves, these powerful environments are supporting more of the multimedia objects we have highlighted in this text. As is also true for blogs, teachers generally have a choice of templates and, for advanced users, some ability to tweak the appearance and content of the website itself.

Elements of an Effective Educational Website

Visual web page editors and online templates offer the teacher a wide range of colors, patterns, clip art, fonts, animations and even sounds from which to choose. It is tempting, because it is fun, to "liven up" a classroom web page. It is generally true that students are drawn to "bells & whistles" and their use can capture a student audience. However, the wise teacher practices restraint so that content and navigation are not overshadowed.

There are some basic guidelines for educational web page design and content:

- Use a "homepage" to organize your content clearly. This page should contain a prominent title, clear navigation to sub-sections, your contact information, a link to your school's website, current important announcements (if any), and the date the page was last updated (your copyright). Some teachers include a "grabber," such as a word, fact or challenge of the day. If you don't want anyone to download or copy your content, this is the place to make that statement.

²¹ Some open source WIKI applications are also available, an even better solution for education, but one which requires the oversight of a good and experienced IT staff.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

- Organization is key. Make a concept map or organizational diagram²² before you begin the website. There are many ways in which your content can be organized, but it should be navigable by "bullet words" such as Homework, Web Links, Announcements, Project A, Topic B, Class C, Trips, For Parents, Activities, etc. Fig. 11.2 illustrates a common website organization plan for an elementary teacher.

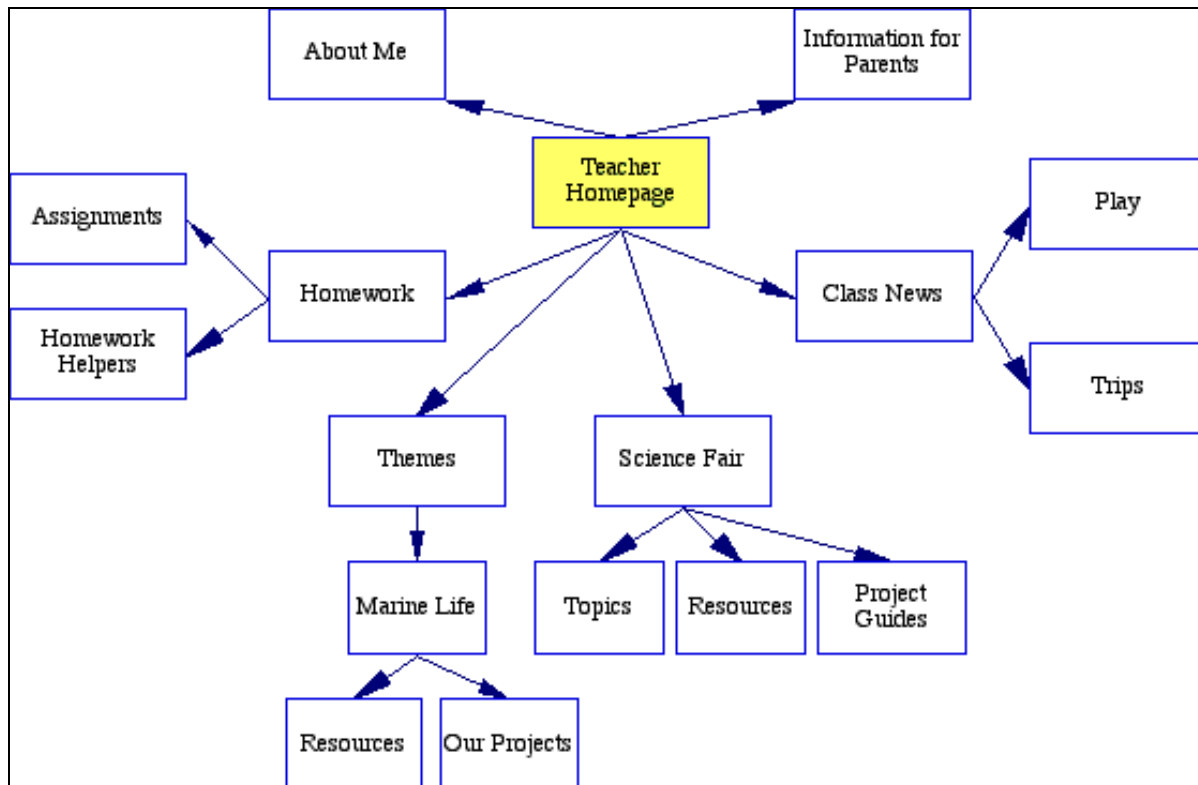


Fig. 11.2 Inspiration map of a teacher website

- Provide the visitor with navigation links to and from every page. Follow the rule of "3 Clicks and 30 seconds" - content should be accessible from any page with no more than three clicks of the mouse and in 30 seconds or less.
- A web page should be up-to-date and useful. If you are directing students or parents to websites, make sure the links work and the URLs are still active. Select the best; limit links to a "short list" of five or so and provide helpful annotations. Links to "outside" web pages should open in a "new window" if possible. This makes it easy for students to return to your home site.
- Steer clear of clutter! The eye is drawn first to images and will follow the direction in which they seem to "point" or "flow." Highlight key text with size and color contrast.

²² Remember that Inspiration concept maps can be exported as HTML, creating the skeleton of a website!

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

- Break large content into several pages. Student users do not want to scroll down, so focus your content on what will fit onto the monitor screen.
- Clip art should be there for a reason and it should be consistent in style and size. If it is used for navigation to subsections, be sure that text labels are included and that the same image is highlighted in the section to which it links.
- Sound and animations should illustrate, not distract.
- Pictures should not be large files! Use an image editor to resize them and compress them. If all else fails, put only one or two on a page and provide links to them. Many teachers post classroom pictures a free online picture gallery, such as *Shutterfly*, rather than putting them on a web page. Others use tools included with photo software such as *iPhoto* to create a quick, efficient web-based slideshow.
- If you want users to download documents (assignments, study guides, public domain text), make sure that they are available in a format that most, if not all, users can open. This would include .txt, .rtf and .pdf files (provide a link to the download site for Adobe Acrobat Reader). Compress large files if possible with *WinZip* or *Stuffit*.
- If you include *Quicktime* or other movies on your site, include instructions for viewing them and an indication of the file size.
- Color schemes and fonts should be consistent throughout the site. This helps your students to stay on track. Select and size your font so that it is readable by your student users²³ and viewable on all computer platforms. Avoid dark backgrounds and background images (except perhaps on your homepage).
- Copyright of images, text, audio and video must be respected. Fair Use does not apply to publicly accessible web pages. Remember that you are modeling ethics for your children; when in doubt, either get (and note) permission or don't use the content! This includes content, such as poetry and artwork, used to create student projects posted on your pages.
- Do not include student names anywhere in the site. Pictures of students should also be avoided unless your school has a legal parent/guardian "permission to use pictures" document on file. Keep this precaution in mind if you are sharing student-created projects or videos, which may contain names in authorship or "cast" lists.
- Remember that student work is also copyrighted. You must have the written permission of the student to post work on the Internet. Student work should also include a copyright statement. A recommended statement is: © 2004 XStud, student at My High School (First initial + four letters of last name).
- Be sure to include an "About Me" page. Introduce yourself to parents and visitors here, tell about your education, experience and philosophy and include information that

²³ San serif fonts, such as Verdana and Arial, are easier for students with disabilities to read. Similarly, text and background color should contrast. Black text on a cream-colored background is visually ideal.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

validates you as a "professional expert" and interesting person. If you have a pet or favorite hobby or interest, you might want to highlight it here.

- Double check your spelling and your facts!

By all means use a website that you like as a model. It is also handy to use a web page evaluation rubric as a guide for designing your own page. Good rubrics, along with other useful resources for web page design, can be found at Kathy Schrock's Guide for Educators: <http://school.discovery.com/schrockguide/eval.html>.

AUTHORWARE ESSENTIALS

In Chapter 10 we looked at hardware and software tools for the creation and manipulation of multimedia and hypermedia. It is important for teachers to remember that good courseware is composed of many elements that work together to support and encourage learning; it is this rich environment that sets digital courseware apart from texts and reference books. The tools we have touched briefly upon in this chapter can stand alone, but they are greatly enhanced by the thoughtful use of other media authoring and editing tools.

Putting it all together, you should recognize by now the value of learning to use a few basic hardware and software tools with confidence: an image editor/paint program, a sound recording/editing application, one or more of the productivity tools highlighted in this chapter, a digital camera, and a scanner. You must also develop a basic skills set for multimedia authoring:

- Internet searching, URL bookmarking and capture, and file downloading,
- Use of the network and other options for file storage,
- Organization files and folders on your storage medium,
- Understanding of file naming and file extensions (digital file formats),
- Ability to use the menus and icon-based toolbars that most applications have in common,
- Storyboarding or concept mapping (digitally or on paper),
- A willingness to try new things.

As is true of all teaching all of the time, content must focus upon the standards and goals, and upon the abilities of the students. It is possible for teachers to create masterful multimedia courseware that is entirely over the heads of the students; it is equally possible for teachers and students to create media rich courseware that is empty of educational content²⁴. Careful planning and a process of review and revision must, therefore, be part of all authorware use.

Even when all systems are GO, none but the most dedicated teachers can be expected to spend the time it takes to create quality courseware content unless school districts make it worth their while. The allocation of release time and money is an important pre-requisite to promoting this worthwhile work. An enterprising school district will gather together subject area and age group

²⁴ The word "PowerPointless" has been coined by Jamie McKenzie to describe such projects.

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

experienced teams of teachers to work, perhaps over the summer, to develop Web-based materials that can then be used by teachers not only throughout the district, but anywhere in the world where similar curriculum standards and content pertain.

LOOKING BACK

It is often difficult for gung-ho computerphiles²⁵ to appreciate the technological timidity of some of their colleagues. The same problem arises when teachers find it difficult to appreciate the lack of academic motivation on the part of some students. Whatever the reason, it is a fact of life that must be accepted and dealt with through patience, understanding, unwavering support and encouragement. The variety of systems being introduced in schools today, exciting as they are to some, can be overwhelming to others. But, since the schools exist for the sake of the students, the teachers--whether computerphiles or computerphobes--must come to terms with the technology to some degree in order to provide an adequate educational opportunity for the students and, increasingly, adequate communication with the parents.

For many teachers, developing lessons around multimedia systems will not go beyond the simplest application of the technology unless and until comprehensive multimedia curriculum materials are available in an easily adaptable format to meet local approaches and the needs of local populations. Canned multimedia applications will generally not work for all but the minority of cases. For this reason, schools will in the long run need to tap the resources of their own technology-oriented teachers--the computerists--to train, and above all work, with those teachers who are naturally and understandably reticent about authoring multimedia systems on their own.

In the end, the students who have grown up around technology, and consequently are not afraid of it, will be the most avid users of the multimedia and other learning systems. Given the opportunity, these students will use the technology to access and assimilate the knowledge they need to achieve educational goals.

But children cannot do this without help. Teachers in the Information Age will need to be able to use technology in support of teaching, and, above all, they will need to know how to establish an environment in which students can safely and successfully control their own learning.

LOOKING FORWARD

Writing one's own software for computer-assisted instruction or multimedia learning is all well and good, and if you can do it, it will save money. But it will not buy the equipment, nor will it generate the funds to release you from teaching so that you can have the time to develop the software!

Money is by no means the only solution to the problem of restructuring schools to meet the needs of an Information Age, but there is precious little that can be done without it. Millions of dollars is available to fund technology projects in schools. The money is offered by foundations and

²⁵ A computerphile is a lover of computers. Its opposite is a computerphobe. Which are you?

EDUCATION FOR AN INFORMATION AGE

Teaching In The Computerized Classroom, 6th edition

Copyright © Bernard John Poole, Betsy Sky-McIlvain, Lorrie Jackson, Yvonne Singer, 2006, all rights reserved

Chapter 11: Creating Computer Applications for Education Environments

other grant-giving institutions at the federal, state, and local levels. Some schools are making great efforts to win those grants. As a result, their students are benefiting from the best that educational technology has to offer.

It should be of great interest to teachers and administrators alike to learn how to effectively apply for grants. It will also be useful to discuss what steps should be taken to get the most out of a grant that has been won. Chapter 12 will therefore look at all aspects of writing grants.