Educational Software Cooperative

The Best of the ESC Newsletter

Print This Form

EDUCATIONAL SOFTWARE COOPERATIVE MEMBERSHIP APPLICATION

YES! I'D LIKE TO JOIN NOW:

___Basic $25 ___Contributing $50 ___Supporting $100 ___Lifetime $1000
(Amounts are in U.S. dollars. Membership is renewable each April.)

Name: _____

Company Name: _____

Address: _____

_____

_____

Email: _____

Phone: _____

Fax: _____

BBS: _____

PLEASE CHECK ONE:

____ Author / Developer (Do not include software with your application)

____ Software Publisher / Vendor / Distributor

____ BBS Sysop / Online Provider

____ Teacher / Administrator

____ Parent

____ Other:_____

MAIL TO:

Educational Software Cooperative
11846 Balboa Blvd, Suite 226
Granada Hills CA 91344
USA

___Check ___Money Order -- _____Amount Enclosed
Prices shown are in U.S. dollars. We regret that we cannot accept checks drawn on non-U.S. banks.

Credit card orders: ___Visa ___MasterCard ___American Express

Credit card number:_____ Exp. Date:_____

Name exactly as shown on the card:_____

Signature:_____

Credit card payments may be placed by phone or fax to Pik a Program:

voice: 212-598-4939
fax: 212-228-5879

If you received this form after December 1996, please contact us for updated information.

EDUCATIONAL SOFTWARE COOPERATIVE

Educational software is one of the most popular and fastest-growing categories of software. In 1992, the Educational Software Cooperative (ESC) was founded by Andy Motes, author of the award-winning "School Mom" program.

Incorporated in 1994, ESC is a nonprofit organization whose purposes are to advance the mutual benefit of authors, publishers, dealers and distributors of educational software; to voluntarily cooperate formally and informally with each other to better develop, advertise, distribute and sell educational software; and to provide the public with information regarding the benefits, uses, and availability of educational software.

Software users, educators, developers, publishers, distributors, and others work together to promote these goals.

A wide variety of educational shareware titles for both children and adults are available on a CDROM,CD-ROM which can be purchased through many of the Author and Vendor members. We expect to produce a new CD approximately every six months.

We currently publish a bimonthly newsletter containing articles written by ESC members and associates, along with product reviews, tips and suggestions for better ways to write and market software and discussions of educational issues. In this newsletter, you'll also find a list of new program additions and updates.

Since 1993, the ESC has had its own section on CompuServe where its members can communicate directly with each other and the public. You can participate in this section whether or not you are a member of ESC. Just enter GO EDFORUM at any CompuServe prompt, and join us in Section 2.

We now have our own Home Page on the World Wide Web:
http://members.aol.com/edsoftcoop

Our activities have included participation in trade association events in Indianapolis and Atlanta. This year, we will present the annual People's Choice Award for Best Educational Shareware at the Shareware Industry Conference in Scottsdale. We work with other organizations on issues, such as software ratings, that affect the industry as a whole.

All officers are unpaid and elected by the membership. As with any organization, there are costs involved, requiring a membership fee to be collected on a yearly basis. Basic annual membership dues are $25, and include a newsletter subscription and one copy of the CD-ROM. Contributing ($50), Supporting ($100), and Lifetime ($1000) members receive additional CDs and special recognition for their important contributions to our ongoing efforts. Membership is renewable each April.

- Membership Application
- ESC CD-ROM
- ESC Bylaws
- Main Menu

JOIN NOW!

Don't put if off any longer! It's never too late to start your ESC Membership and participate in our ongoing activities:

- Our own Home Page on the World Wide Web, with links to ESC members' FTP sites and home pages for this service: http://members.aol.com/edsoftcoop

- A "Members Only" Home Page with special information and links of special interest to ESC members

- Our Member of the Month project, featuring a different ESC member's work each month

- The ESC CD-ROM is better than ever, with more educational categories, and room for members' non-educational applications, too

- Sponsorship of the People's Choice award for Best Educational Shareware, presented at the annual Shareware Industry Conference (SIC)

- Our annual meeting and luncheon at SIC

- Our bimonthly newsletter

- Our discussion section and library on CompuServe: GO EDFORUM

- Cooperation with other associations on issues that affect us all, such as software ratings

- In the works - an online software catalog

- Your chance to interact with members who share your interests in meeting the special challenges of writing, distributing, and locating educational and "edutainment" software

- Membership Application
- ESC CD-ROM
- Main Menu

ESC CD-ROM 

The ESC CD-ROM and many other fine CD-ROM shareware collections are available from:

Most Significant Bits
37207 Colorado Avenue
Avon, OH 44011
Voice: 216-934-1397, 800-755-4619
Fax: 216-934-1386

WHAT IS SHAREWARE?

Shareware is an exciting marketing method that allows you to try top-quality software before paying the author. The small fee you pay to a disk vendor covers only the duplication and distribution costs, and permits you to evaluate these programs. If you continue to use a program, you must send the author an additional payment, which may entitle you to technical support, a printed manual, bonus programs and more. Your payment supports the authors, enabling them to continue writing newer and better Shareware programs for you.

ESC OFFICERS

Rosemary West, President	74774.403@compuserve.com
John Gallant, Vice President	76170.2251@compuserve.com
Richard Hart, Treasurer	71222.3536@compuserve.com
Bert Fischer, Secretary	70233.1315@compuserve.com
Andy Motes, Board Member	73757.1111@compuserve.com

BYLAWS OF EDUCATIONAL SOFTWARE COOPERATIVE, INC.

ARTICLE 1: OBJECT OF CORPORATION

Section 1. The Educational Software Cooperative (the "Corporation") is created for the following purposes, to the extent that these are not inconsistent with the New Jersey Nonprofit Corporation Law and the Corporation's Articles of Incorporation:

To advance the mutual benefit of authors, publishers, dealers and distributors of educational software; to voluntarily cooperate formally and informally with each other to better develop, advertise, distribute and sell our educational software; to provide the public with information regarding the benefits, uses, and availability of educational software.

Section 2. The purposes of the Corporation may not be altered except as an amendment to these bylaws, which shall require a two- thirds majority vote at an annual or special meeting of the Membership.

Section 3. The Corporation is not organized, nor shall it be operated, for pecuniary gain or profit, and it does not contemplate the distribution of gains, profits or dividends to its Members and is organized solely for nonprofit purposes. The property, assets and profits and net income of the Corporation are irrevocably dedicated to the purposes set forth in Section 1 hereof, and no part of its profits or income shall ever inure to the benefit of any Director, Officer or Member thereof or to the benefit of any private person. Nothing in this section shall be interpreted to prevent the Corporation from hiring and paying employees, nor shall the Corporation be prevented from paying any individuals or organizations for professional services, or from purchasing materials needed to conduct the Corporation's business.

ARTICLE 2: MEMBERSHIP

Section 1. The Membership of the Corporation shall consist of those persons who have signed the Certificate of Incorporation as incorporators together with all persons who are hereafter received in or elected to Membership as hereinafter provided.

Section 2. The criteria for Membership in the Corporation shall be:

(a) That the Member agrees to uphold the purposes of the Corporation as defined in the bylaws.

(b) Members must pay and remain current in the payment of dues, fees, and assessments to continue as Members.

(c) Memberships belong to individuals, not organizations, and are not transferable.

Section 3. The criteria for Membership in the Corporation may not be altered except as an amendment to these bylaws, which shall require a two- thirds majority vote at an annual or special meeting of the Membership.

Section 4. It shall be the obligation and responsibility of each Member to advise the Board of Directors if the Member no longer qualifies as a Member. A Member shall be dismissed from Membership upon the failure of the Member to meet any Membership criterion, or because of the commission of an act believed by the Board of Directors to be detrimental to the best interests of the Corporation. In the case of a dismissal for a detrimental act, the Member shall first be given the opportunity to answer such charges before a meeting of the Board or the Membership Committee. If the Board of Directors votes to cancel a Member's

Membership, they shall provide thirty days' written notice of such cancellation to the Member. In the case of dismissal for failure to pay dues, fees or assessments, the Member may be automatically reinstated upon payment of the amount in question. In the case of dismissal for any other reason, the Membership may vote to reinstate the Member by majority vote at the next special or annual Membership meeting. A member who has been dismissed for any reason other than failure to pay dues or fees may not reapply for membership within one year of the dismissal.

All Membership cancellation actions shall require a majority vote of the Board of Directors. This authority may not be delegated.

Section 5. A Member may resign from Membership at any time and shall be required to do so if such Member is unable or unwilling to comply with Membership requirements.

Section 6. The Board of Directors may establish a Membership Committee to which it may delegate any responsibility which the Board of Directors may have regarding Membership. A Member may appeal to the full Board from any adverse decision of the Membership Committee.

Section 7. Dues, fees and assessments shall be established by the President and ratified by majority vote of the Membership at any meeting.

Section 8. Membership may be held by individuals, companies, partnerships or corporations. If the membership is held by a company, partnership or corporation, one person from that organization shall represent it, participate in meetings, and vote.

ARTICLE 3: GOVERNMENT

Section 1. The general management of the affairs of the Corporation shall be vested in the Board of Directors who shall be elected as provided in the bylaws. Members of the Board of Directors must be Members of the Corporation.

Section 2. There shall initially be four (4) Members of the Board of Directors. The Board of Directors or the Membership may set the size of the Board of Directors, to take effect at the next election for the Members of the Board of Directors. A majority of the Board of Directors may not be employed by the same organization and/or by organizations that substantially control the employing organizations of other Board Members.

Section 3. The term of office of each Member of the Board of Directors shall be two (2) years, or until the Member's successor is elected.

Section 4. Members of the Board of Directors shall be eligible for reelection.

Section 5. The Board of Directors shall meet prior to each Annual Meeting of the Membership, and may meet from time to time as deemed necessary by the Members of the Board. The Board may choose to hold its meetings on an electronic forum such as HTTP,CompuServe or such other forum as may be selected by the Board of Directors. All Directors must have access to the electronic forum that is the site of such electronic meetings.

ARTICLE 4: MEETINGS OF MEMBERS

Section 1. Annual Meetings of the Members of the Corporation shall be held once each year

at a time to be fixed by the Board of Directors. Final and official notice of the time and place of the Annual Meeting shall be provided to each Member not less than ten nor more than fifty days prior thereto and shall specify the matters to be discussed and voted upon. No business may come before an Annual Meeting which is not so specified. The board may choose to hold the Annual Meeting electronically. Members may be present at an Annual Meeting in person or by written or electronic proxy.

Section 4. Special Meetings of the Members of the Corporation may be called from time to time by the Board of Directors, or by at least 10% of the Membership acting in concert, or by at least 15 Members acting in concert, or by the Secretary as specified in Article 6. Members shall be deemed to have acted in concert for purposes of the preceding sentence if they have provided written notice to the Secretary of the request for a Special Meeting, such request to specify the matters to be addressed at such meeting. Notice of the time and place of a Special Meeting shall be provided to each Member not less than ten nor more than fifty days prior thereto and shall specify the matters to be discussed and voted upon at such Special Meeting. No business may come before a Special Meeting which is not so specified. Special Meetings may be conducted on an electronic forum or bulletin board system ("BBS") chosen by the Board of Directors.

Section 3. At any meeting of the Members, each Member shall have one vote. Members of the Board of Directors shall not have the right to vote on matters concerning the manner in which they have exercised their functions, except they may vote on any matter concerning the description, enlargement or circumscription of their functions.

Section 4. At all meetings, a quorum shall consist of those persons who have cast their votes at such meeting.

Section 5. Action at any meeting of Members may be taken by a simple majority vote of a quorum, except as to any requirements for a super-majority vote specifically set forth in these bylaws.

Section 6. Members who are unable to attend an Annual Meeting may designate a proxy by written notice to the Secretary on the matters on the agenda, appointing the Board of Directors to cast votes for such Member in the manner specified in such proxy.

Section 7. The President shall chair all meetings. In the absence of the President, the chair shall pass to the remaining Officers of the Corporation, in the order they are named in Article 7. The meetings shall be governed by Roberts Rules of Order, Revised (1979 edition) except where, in the opinion of the chair, a limitation or enhancement of electronic conferencing makes certain of those rules either unworkable or unnecessary.

Section 8. Any resolution which is defeated at any meeting may not be reintroduced or placed on the agenda for any meeting within six (6) months following defeat of such resolution.

ARTICLE 5: PROCEDURE FOR MEETINGS

Section 1. Only the meeting chair may call for an end of discussion and for a vote on a proposal. In the case of an electronic meeting, such call shall constitute the beginning of the "voting period".

Section 2. In the case of an electronic meeting, the voting period for any issue or election shall be 168 hours (seven days). Should the BBS which is used for the meeting be unavailable to the general Membership for six or more continuous hours during the voting

period, the voting period shall be extended for an additional 24 hours.

Section 3. In the case of an electronic meeting, in lieu of voting on the BBS, a Member may send a written vote to the Secretary or other designated person and it shall be counted if received before or during the voting period. In the case of a non- electronic meeting, in lieu of voting in person, a Member may send a written or electronic vote to the Secretary or other designated person and it shall be counted if received prior to the call for a vote.

ARTICLE 6: ELECTION OF DIRECTORS AND OFFICERS

Section 1. The Directors of the Corporation shall be elected at the Annual Meeting.

Section 2. No less than fifteen (15) days prior to the Annual Meeting, the President shall appoint a Nominating Committee for the purpose of nominating candidates for the Board of Directors. The names of the nominees shall be included with the notice of the Annual Meeting sent to members as provided in Article 4, Section 1. Additional nominations by the membership will be accepted from the floor during the Annual Meeting. The requirements of this section will not be in effect for the first Annual Meeting of the Corporation

Section 3. Following the election of Directors, the Directors shall elect from the Members a President, a Vice-President, a Secretary and a Treasurer.

Section 4. If a vacancy occurs among the Board of Directors, a majority or the remaining Directors shall appoint a new Directors to fill the vacancy until the next annual or special meeting of the Membership.

Section 5. If a vacancy occurs among the Officers, the vacancy shall be filled by the Board of Directors from the Membership.

Section 6. In the event of a tie vote by the Board of Directors for any Officer, a vote will be cast by the first of the President, Vice President, Secretary, or Treasurer that is not a member of the Board of Directors.

ARTICLE 7: DUTIES OF OFFICERS

Section 1. The President may establish committees and shall appoint heads of such committees. The President shall act as Chief Executive Officer of the Corporation, to coordinate the activities of the Officers and the committees and shall provide guidance and leadership in the day-to-day operation and functioning of the Corporation. In the absence of the Treasurer, the President shall perform the Treasurer's duties.

Section 2. In the absence of the President, the Vice-President shall perform the President's duties. In the absence of the Secretary, the Vice-President shall perform the Secretary's duties.

Section 3. The Secretary shall keep the minutes of all meetings of the Members and of the Board of Directors, shall keep a register of the Members, and shall provide notices of meetings of the Members. The Secretary shall sign the record of meetings.

Section 4. The Treasurer shall keep accurate books of account, prepare and present periodic operating statements and balance sheets to the Board of Directors, and deposit and withdraw funds of the Corporation under the direction of the Board of Directors.

Section 5. Any Officer may be removed from office for cause by a two-thirds majority of either the Board of Directors or the Membership at any meeting called for that purpose.

ARTICLE 8: DUTIES AND POWERS OF THE BOARD OF DIRECTORS

Section 1. The Board of Directors shall have general charge and management of the affairs, funds and property of the Corporation. They shall have full power and it shall be their duty to carry out the purposes of the Corporation according to its charter and bylaws; to determine whether the conduct of any Member is detrimental to the welfare of the Corporation and to fix the penalty for such misconduct or any violation of the charter or bylaws; to employ personnel for the carrying out of the Corporation's objectives; and to make rules for the conduct of the Members.

Section 2. Any action required or permitted to be taken by the Board of Directors may be taken without a meeting if all Members of the Board consent in writing to the adoption of a resolution authorizing the action.

Section 3. Meetings of the Board may be called and governed in such manner as the Board may from time to time determine.

Section 4. A quorum of the Board shall ordinarily consist of 66% of the Members of the Board. In the case of an electronic meeting, if a motion has been open for voting for one week, a quorum shall consist of 50% of the Board for the purpose of that particular motion.

Section 5. Any Member of the Board of Directors may be removed from office for cause by two-thirds majority of the Membership at any meeting called for that purpose.

ARTICLE 9: INDEMNIFICATION; INSURANCE

Section 1. The Corporation shall indemnify and hold harmless from all costs and expenses (including reasonable attorneys fees) of any person who was or is an elected or appointed Officer or director of the Corporation and is threatened to be or has been made a party to an action, claim, or other proceeding arising out of such person's performance, purported performance, or failure to perform, any duties on behalf of the Corporation. Such indemnification shall not extend to liabilities arising out of a person's gross negligence, misfeasance or willful misconduct.

Section 2. The Board of Directors is authorized to obtain Directors and Officers liability insurance to shield such persons from liability for all costs, expenses and attorneys fees arising out of the conduct of their duties as Directors and Officers, except for liabilities arising out of their gross negligence, misfeasance or willful misconduct.

ARTICLE 10: DISSOLUTION

Section 1. The Corporation can be dissolved only upon a two- thirds majority vote of a quorum present at any Annual or Special Meeting. On dissolution or winding up of the Corporation its assets remaining after the payment of, or provision for the payment of, all debts and liabilities shall be distributed as determined by the Board of Directors of the Corporation. If the Corporation holds any assets outside the state of its incorporation they shall be disposed of as required by law.

ARTICLE 11: NOTICES AND COMMUNICATIONS

Section 1. All notices or communications required or permitted hereunder may be sent by first-class mail or by electronic means. Such notices or communications shall be deemed to be delivered upon deposit with the United States Postal Service, or upon submission via the electronic means designated for such purpose. All notices and communications shall be addressed to each person at the last known address shown in the corporate records.

ARTICLE 12: AMENDMENTS

Section 1. These bylaws may be amended only by a majority vote of those voting at an Annual or Special Meeting provided that notice of the purpose of any proposed amendment has been stated in the call for the meeting.

ARTICLE 13: FISCAL YEAR

Section 1. The fiscal year of the Corporation shall be as determined by the Board of Directors.

ARTICLE 14: SEAL AND CORPORATE EMBLEM

Section 1. The Corporation may have a seal as adopted by the Board of Directors. The Seal may be used by the Officers to attest to the documents of the Corporation.

Section 2. The Corporation may have a corporate emblem as adopted by the Board of Directors. The corporate emblem may be used by Members under guidelines established by the Board of Directors.

ARTICLE 15: AUTHORIZATIONS

Section 1. Contracts. The Board of Directors may authorize any Officer, or may authorize any Officer to delegate such authority, to enter into any contract or execute and deliver any instrument in the name of and on behalf of the Corporation. Such authority may be general or confined to specific circumstances.

Section 2. Checks, Drafts, Etc. The Board of Directors may authorize any Officer, or may authorize any Officer to delegate such authority, to issue checks, drafts, or other orders for the payment of money, notes or other evidence of indebtedness in the name of the Corporation.

Section 3. Deposits. All funds of the Corporation shall be deposited from time to time to the credit of the Corporation in such banks, trust companies or other depositories that the Board of Directors may select.

Section 4. Gifts. The Board of Directors may accept on behalf of the Corporation any contribution, gift, bequest or device for the general purposes or for any specific purpose of the Corporation.

- Membership Application
- ESC Background Information

- Main Menu

Visit our ESC Home Page on the World Wide Web:
http://members.aol.com/edsoftcoop

On CompuServe, chat with us in Section 2 of
EDFORUM

DO KIDS WANT TO LEARN?

Bob McElwain

It's been said that somewhere along about age eight, nine or ten, kids lose all interest in learning. Nuts! It fades, though. And sometimes it's hard to find even a trace. But it's there. In every kid in the land. The trick is to latch on to it. If you can, neat things happen. Conversely, if you don't, you're wasting time. There'll be no involvement on the part of the kid. Hence nothing of significance learned.

As a teacher I've got to get it done. Fan a spark into a flame, so to speak. Then, of course, I've got to make sure my students get good stuff. Useful skills and ideas. As an author of educational software, I face an additional challenge, for my "student" is not mandated by state law to attend my "class."

Authors know kids are bombarded with slick polished presentations. When thinking of pre-schoolers, Sesame Street comes to mind. Some thus feel they must compete at this level for the kid's attention. I'm don't think it's so.

A sharp, jazzy opening may be just the ticket. But this sort of thing does not need to be sustained to any significant degree. Certainly it must not be the whole of it. Else we're kidding all concerned by leading them to believe that learning is one grand video game after another.

In the evaluations of educational software done by "PC Magazine", fun is a heavily weighted factor. I personally object to the word when applied to learning. Enjoyment is good; it happens now and then. Practical and/or rewarding are better. But I surely don't feel fun is as significant a factor as some do.

On more than one occasion, I've been forced to pretty much ignore a group of students for extended periods. Initially, all slack off. Then, one by one, most will get back to assigned tasks. And if there's been good solid introduction, and the subject is perceived by the students as practical or necessary, the goofing off is minimal.

I am convinced that given an appropriate introduction, which includes a demonstration of the worthiness of the subject, most kids are interested in learning. While the degree varies, of course, interest does exist. And it can be sustained without bells and whistles. In fact, slick tricks can hinder, rather than help.

Vol 4, No 5, November 1995

- Main Menu

STANDARDS FOR EDUCATIONAL SOFTWARE

Marilyn Brown

Since one of the goals of the ESC is to develop standards for educational software, I would like to contribute a summary of standards that were developed in Canada by the Council of Ministers of Education for evaluating educational software (Software Evaluation Criteria for educational computer software evaluation, 1985). This may save a lot of time by providing a framework for discussion.

Objectives:

The questions to be answered are:

* Are the objectives of the program clearly stated?
* Are the objectives appropriate to the target audience and the medium?
* After satisfactory completion of the program will the objectives be fulfilled?

Pedagogical Content:

The questions to be answered are:

* Is the pedagogical content appropriate to the target audience?
* Has the pedagogical content been effectively designed?

1. Scope (Breadth, Range)
   a. Is the scope appropriate to the target audience?
   b. Are the objectives appropriate to the medium?

2. Sequence
   a. Is the content sequence appropriate to the target audience?
   b. Is the content sequence effectively designed?

3. Depth
   a. Is the content depth appropriate to the target audience? (amount of instruction and practice appropriate, content level relevant to the interests and learning level)
   b. Is the content depth effectively designed? (definitions and explanations available, appropriate amount of detail, sufficient number of examples)

4. Accuracy
   a. Is the content accurate?
   b. Has a simulation model been accurately represented?

5. Bias
   a. Does the content contain biases? (ethnic/racial references, social class references, violence, age portrayals, political references, social role references, stereotyping, unfair/inaccurate judgments)

6. Readability
   a. Is the general readability/reading level of the content appropriate for the target audience?

Instructional Format:
The questions to be answered are:

* Is the instructional format appropriate to the target audience?
* Have the capabilities of the computer been effectively utilized?

7. Student Interaction
   a. Is the interaction appropriate to the target audience? (can students interact easily, are there sufficient instructions, is the program tolerant in accepting unexpected inputs)
   b. Does the program contain a method of inquiry that promotes learning? (does student learn through manipulation of content rather than by passively reviewing facts)
   c. Is the interaction effective? (does the interaction promote learning, is there sufficient interaction)

8. Questioning Technique
   a. Is the questioning technique appropriate for the target audience?
   b. Is the questioning technique effective? (questions appropriate to the content, questions effectively randomized where necessary)

9. Feedback
   a. Are the form and content of the feedback appropriate to the target audience?
   b. Are user inputs accurately evaluated as right or wrong? Does the program distinguish between the wrong answer and the wrong format?
   c. Does learning take place regardless of the student's response?
   d.Is feedback non-threatening, immediate, positive, motivational and user sensitive?
   e. Are cues/prompts used after a wrong response?
   f. Is corrective feedback provided?
   g. Is the feedback relevant to the user's history of responses?
   h. Is negative feedback unnecessarily attractive?
   i. Can the user/teacher control the feedback where appropriate?
   j. Is quantitative (summary) feedback provided?
   k. Is quantitative feedback effective?

10. Evaluation
   a. Are pre-tests/post-tests appropriately present/absent?
   b. If present, are pre-tests/post-tests effectively designed?

11. Branching
   a. Are branches appropriately present/absent?
   b. If present, are branches appropriate to the target audience?
   c. If present, are branches effective in accommodating individual differences?

12. Types of Control (Program, Teacher, User)
   a. Does the user/teacher have an appropriate amount of control over the program?
   b. Are the control features which are present effectively designed?

Technical Design:

The questions to be answered are:

* Is the technical design appropriate to the target audience?
* Have the technical capabilities of the computer been effectively utilized?

13. Screen Displays

a. Are the character size, font and case appropriate to the target audience?
b. Are the screen displays effective? (free from grammar, spelling, punctuation and hyphenation mistakes, amount of material presented at one time, material clear and easy to read, special features used appropriately, transition from display to display smooth, well paced and unobtrusive, appropriate amount of time for user to read and absorb the information)

14. Colour, Graphics and Sound
    a. Is the use of colour, graphics and/or sound appropriate to the target audience?
    b. Are colour, graphics and/or sound used effectively? (add to the effectiveness of the instruction, can sound be controlled, is use of features motivational, do graphics portray intended object/idea, are quality and clarity appropriate)

15. Ease of Use
    a. Can the intended user easily and independently operate the program? (exit the program, return to the menu, move to another section, are instructions available at appropriate points and are they effective, is program reliable)

Implementational Support:

The question to be answered is:

* How easy is it to use the software in the classroom?

16. Ease of Implementation in the Classroom
    a. Are sufficient user/teacher support materials available?
    b. Are they effective?
    c. How difficult/easy is it to implement the software in the classroom? What additional demands are placed on the teacher's time to use the package and to operate it within a classroom setting?

17. Management System
    a. Are there sufficient management functions available?
    b. Are they effective? (may include class lists, reports, prescription, diagnosis of student's weaknesses, security, capacity)

Summary:

18. Objectives
    a. Are the developer's objectives clearly stated?
    b. Are they appropriate to the target audience?
    c. Are they appropriate to the medium?
    d. Will satisfactory completion of the program result in fulfillment of the objectives?

19. Summary Statement
    a. What are the major strengths and weaknesses of the software?

MEANING-MAKING
for Class Notes and Assigned Readings
Richard A. Hart, PhD

Once the skill of meaning-making has been learned, a student becomes a self-correcting scholar, the joy of both parents and teachers. A student who learns and reports with confidence. A student who feels good about self and what he or she is doing. A student who takes pride in practiced independent self-judgment. A student who can handle school regardless of course, class, or special exam.

Students can learn by memorizing what they are told or read. This is the concrete level of thinking, the "what you see is what you get" level of thinking. Higher levels of thinking require the student to "do something" with this information or observation. To question it. To make it meaningful. Only the student can do this. (A teacher can assist by creating the learning environment for meaning-making, but cannot interact with the internal process unless the student verbalizes in a one-on-one tutorial session. The student talks. The teacher listens.)

Meaning-making extends over the full range of levels of thinking from random-guessing to formal problem-solving. For class notes and assigned readings, it can be divided into four steps of list, describe, relate, and use.

1. Look through the notes or reading and make a LIST of "important" terms. These are terms, of one or more words, that you feel you need to be familiar with to understand the assignment. You can make two lists, terms you understand and can use, and those you do not.

2. Describe the action, thing, condition, or observation for which the term is a label. This is the reverse of defining a term where you often end up with a meaningless, useless definition for an unknown term: a matched set of nonsense. To use a term as a label, you must first describe the thing or condition, or be able to visualize it. If you can claim actual or imaginary experience, you can then claim you know and can use the term as a label for that experience. If not, you need to read the assignment again or other sources, or ask questions in class, at home, in laboratory, or on field observations.

3. Relate the terms by putting them into an order. This is often called word mapping. Terms that are labels for similar things, that have the most to do with one another, are placed close together.

4. Use or express your understanding with appropriate limits. Writing a paragraph makes a good example. The opening sentence can contain the description for the label. Follow this with statements that set the limits of the description. Close with a summary statement or example of what the term is a label for.

The paragraphs can be reviewed to refresh memory, if needed, when test time comes. Things that are understood do not require the repeated memorization required by things that are just cold memorized (where the loss is about 50% every 24 hours). Experience and understanding are much more permanent.

The sequence is: list, describe, relate and use ("do something with"). This sequence developed during empirical research on how to use computers to assist struggling students

to become self-correcting scholars.

The computer program, Expeditor, accepts a paragraph in the form of a question. The opening sentence is the question stem. The limiting statements (setting outside what is acceptable) are the wrong answers. The summary statement becomes a second right answer.

The program will accept a description as the question stem and the term as the right answer. As your understanding grows, you can enter a second right answer and up to four wrong answers (limiting statements that are not acceptable).

Is the computer needed? No, but it sure helps. Rather than review a paragraph that rests in one form on a sheet of paper, on a computer the information is presented in several ways including short answer, visualized, scientific, multiple-choice and multiple-guess. Also what you have learned at several levels of thinking can be shared with others. Disks can be traded or sold. Question files can be up- and down-loaded onto computer bulletin boards (BBS).

The final step in learning is putting it to the test. Expeditor test bank files can be shared with classmates who can edit, add, and delete. Your teacher can edit them for course tests which can be scored free-choice as well as the traditional forced-choice. This will allow the computer program, Trainer, to score tests by quality and quantity. Did you understand? Did the question perform well and at what level of thinking?

The complete sequence is: list, describe, relate, use and verify. The self-correcting scholar does this mentally, internally. For those needing to practice, a computer program adds needed animation and ease of information handling. Students report they prefer the dynamic computer review to their static written notes. Expeditor is appropriate for any student, Junior High through College, who has access to an IBM or DOS compatible computer. A printer is required for making practice or real paper tests. Question entry is best done with a group of three to five students using paper or a computer.

WRITING ON THE COMPUTER

Rosemary West

I just received a sample copy (October-November 1994) of "Journal of the Electronic Writer", the newsletter of the newly-formed Electronic Writing Users' Group (E/WUG). This non-profit organization was started by the same people who run W/PUG, the WordStar users' group. The group's purpose is to support the use of computers as writing tools, and the newsletter editors hope to provide a good mix of information about computer writing and writing per se.

This issue had 23 very readable pages. It included articles on grammar and punctuation, differences between PCs and typewriters, purchasing high-end computers, using hypertext, identifying authors' styles, speech writing, book design, software reviews, and assorted odds and ends.

Overall, I was favorably impressed. The newsletter held a lot of well-presented information. It would be of use when doing any kind of writing, from the Great American Novel to my next VENDOR.DOC file. It's published six times yearly; membership is $27 in the US, US$32 in Canada and Mexico, US$37 internationally. For information or to join: E/WUG, PO Box 16-1443, Miami FL 33116-1443.

MAGAZINE REVIEW

Rosemary West

Two publications that I find very helpful are Inside Microsoft Windows and Inside Visual Basic for Windows, both published by The Cobb Group. Although they seem pricey ($49 and $59 per year for very slim magazines that sometimes contain only two or three articles per issue), they are filled with valuable technical information, tips and techniques that are extremely difficult to find elsewhere.

As independent publications, they aren't afraid to discuss the flaws and limitations of the software. Because they contain little or no advertising, they avoid the suspicion, provoked by many other computing magazines, that editorial content might be overly influenced by the advertising department.

Inside Microsoft Windows is intended for end users rather than developers, but contains plenty of information that can help programmers, especially those who still find the transition from DOS to Windows uncomfortable. Recent articles included information on deciphering WIN.INI, sizing bitmaps in Paintbrush, creating PIFs for DOS applications, connecting to the Internet, and previews of Windows 95.

Inside Visual Basic For Windows provides detailed information on language features, the use of controls, and programming techniques. Unlike some of the other magazines aimed at VB users, it rarely digresses into third-party add-ons or competing products, but generally sticks to what can be done with VB (or VB Pro) out of the box. Recent articles included information on the 3DPanel control, extracting icons from other applications, aligning numbers, and handling databases.

Both magazines contain plenty of screen shots and code listings. Without "dumbing down", they use plain English explanations, and provide the detail that is sorely lacking in the user's manuals. I have only two real complaints: Sometimes an article from one magazine is duplicated in the other, and I wish each issue contained more articles. Subscription information can be obtained by calling 800-223-8720 or 502-493-3300.

Vol 4, No 3, June 1995

- Main Menu

LET'S FINISH IT!

Bob McElwain

People are well intended. Sure. There's exceptions. But most folks want to do the right thing. And do it well. Kids are like that. Like people, I mean. And doing well helps them feel better about themselves. Something most desperately need.

Some authors don't think in such terms. For example, those who market fun and games as educational. My objection is not with the product. My concern is only with the labeling. It misleads both parents and kids. To the extent that either is led to believe that playing games is educational, to that extent we're lying to them. And that's not nice.

While learning is fun now and then, it's often not. Words such as rewarding and satisfying, better describe the high points. Words like drudgery and painful, come into play for the youngster who's struggling. Like with the multiplication table someone said should have been learned three years back.

Obviously an author's got to sell. And with a profit. Else bankruptcy comes quickly. And if it's games that work, go for it. But if the product is labeled educational, we have quite a different set of responsibilities to both the parent (purchaser) and the kid (user).

If the presentation is professional and the content leads to improved skills or new ideas that are useful, the responsibility to the parent has been fulfilled. Much more is needed, however, before the same can be said for Johnny.

First, he must know what needs to be done. And something of the why of it. For he needs to be convinced it's a "right" thing to do. Particularly tough, if it's something he won't want to do. Then he needs to be shown how to do it. Step by step, if necessary. And he needs to succeed at each step. For then the confidence goes up. And he's more willing to take the next one.

And when he's finished, be sure he knows he has succeeded. For until he does, until he feels better about himself because of this accomplishment, we haven't finished. We've only a birthday cake with lighted candles. What we need to see is the glow in the eyes that comes from blowing them all out.

Tough to prove the above adds to the bottom line. But satisfied kids are willing to move on. And parents who are determined to do what they can, are more likely to reach again for their credit card and phone.

As a personal aside, I've a comment about the multiplication table, long division, and other such topics. I'm fed up with those who sigh deeply and decry "the fact" that Johnny should have learned whatever, two years back. That he's hopelessly behind now. Nuts. A lot of really neat kids will never learn how to multiply. Let alone divide. To more rote drill, my answer's always been, "Let's get Johnny a calculator. I'll show him how to use it. Then we'll get on to some real mathematics." When was the last time you needed to divide 3.948 by 6.23, accurate to three decimal places? If it happened, I bet you used a calculator. I do.

ANIMATION NOTES

John Gallant

Many programs have objects that move around the screen. A large percentage of those programs seem to move only one object at a time. Moving multiple objects simultaneously is just an extension of moving a single object, as long as you keep track of what you are doing. The following notes are an introduction to the theory involved (sorry, no code). These are aimed at people with programming experience but little experience in animation. Many of the authors in ESC already are using these techniques and may be able to suggest improvements or short cuts.

In the simplest case, you would draw a background and then draw an object over it. To make things easy, you could leave the background blank. This is not going to make it in today's market. Today you need an interesting background, multiple objects moving over that background, and possibly a foreground the objects pass behind. On top of all that, you may want to add a mouse cursor. Quite a change, isn't it? But not really that hard.

The things to be concerned about are:

1) keeping track of the order in which the objects are "stacked" on the screen
2) using fast graphic drivers
3) synchronizing to a system clock

And if you are doing a lot of screen writing you should consider:

4) using two screens and toggling between them.

Let me start with a description of a generalized approach to multiple objects and then give a different approach used in many games.

When you place an object on the screen you destroy what is under it. Remember that you do not actually lift the object up and magically find the background still under it. For this reason you must first save a copy of what is under the object you are about to draw. To remove the object you redraw the background where it was originally. Assuming that you do not want to redraw the entire screen (bad idea), you will need to save just enough background to cover the object.

Save the area under the object to memory. If you are willing to work with direct video read and write, a good place to store the background is in unused video memory. In high resolution EGA mode there is some video memory between pages. In low resolution EGA there is enough memory for four full pages of display. Using video memory allows you to move all four color planes with one move instruction. This will depend on the video mode you are using, and you should get a good book on graphics hardware.

Once you have made a copy of what is under the first object, place the first object on the screen.

Now it gets a little tricky. If the objects can overlap, you now want to save the area under the second object (counting up from the background). The reason is that the area under the second object may include the first object. This is necessary only if you want to treat the objects independently. That is, if you want to be able to move the second object without having to worry about the existence of the first object (object-oriented software and all that).

Only you know your application.

Continue up through the stack of objects until you get to the top. Now place any stationary foreground objects as if they were just like the other moving objects. In other words, you will need to save what is under them also. Finally, add the cursor if you are using one.

To do the next frame, start by replacing all the pieces of background in REVERSE order. If you do not replace the background in the correct order, you will find you have little pieces of objects accumulating on your screen. Very tacky!

Once everything is replaced down to the original background, start building your stack of objects in their new positions. There is a fair amount of bookkeeping involved, but the effect is worth it. It is interesting and perhaps a bit depressing that the average computer user thinks the objects are actually just sliding around on the screen and has no idea of the actual mechanics. Perhaps it is an indication of a job well done when you can make it look so easy.

Now for the other, somewhat simpler approach. This requires that all objects near any moving objects be repainted. In the previous method, it was possible to repaint only the moving objects and any objects on top of them. This may sound a little confusing so read how this second technique works and then think about what I just said.

This method is used in many video games, such as Commander Keen. (Author's disclaimer: I did not write Commander Keen, so any statements about how it works are pure speculation. I have used both of the techniques described myself, though, so I know they work.)

Create a bunch of cells of uniform size that will neatly cover the background and form a picture. Think of it as a quilt or a tiled wall. Your background then consists of an array of integers pointing to a list of picture cells. If you choose your cells carefully, you can reuse most of them. There is an added advantage here in the reduction in graphic data that needs to be stored.

If you know which section of your picture was disturbed, you just repaint the cells in that area. Once the background is restored, you can start stacking your moving objects in their new positions. You did not need to save anything under the objects; it was already available. You only need to replace the cells in the area disturbed, not the entire screen. Again, store these cells in unused video memory if possible.

These are two of many techniques. I actually combine aspects of both, depending on the need. You can probably find your own variations.

That covers the first item in the list, keeping track of objects. Now for the second item, fast graphic drivers. When I first started 3-Ball Juggler I tried to use the drivers available in Borland's Turbo C 1.5. I used `getimage()` to capture background and circle() to draw a rudimentary ball. This was on a 20MHz 386. I realized immediately that this was not going to do. I went out and bought a copy of Programmer's Guide to the EGA and VGA Cards by Ferraro (Addison-Wesley). (You might also look at Programmer's Guide to PC and PS/2 Video Systems by Richard Wilton).

I also bought an assembler and started learning how to write assembly code for the IBM PC. Fortunately I had written a fair amount of assembly for other machines already. With the new code I was able to move multicolor balls, hands, and forearms better than 10 times the speed of what I could do with just the available Borland library. I am not suggesting you write your own assembly drivers, although many of you already have. I am suggesting that

you look around for a better library than the one that came with your compiler.

If you write your own drivers, keep in mind the actual architecture of the hardware. Take advantage of byte boundaries when writing to the screen. Make as few video memory accesses as possible. Build your images in regular system memory before moving the images to the screen. In other words, think about the hardware, not just the software.

Now for the third item on the list. Once you are done with your picture, it is time to wait. Yes, I said wait. If you just start repainting the screen again you will find that the speed of your animation is dependent on the speed of the computer you are using at the time. Slower machines will run too slow, fast machines will run too fast. You will need to wait for the computer. If you have very little to move and fast graphics software, you can wait for the video refresh. This comes about 70 times per second in EGA mode and makes for very smooth motion.

If you would like to move more or bigger objects then you will need more time. You can wait for the system clock. This occurs about 20 times per second but is still fast enough to make pleasant looking animation. Some major games with lots of screen activity move as slowly as 10 frames per second. This is about the limit of realistic animation. It will most likely require a special clock driver in your software to replace the clock interrupt. I will leave the subject of clock interrupts to others. There is public domain software available to do this on some bulletin boards. Nels Anderson has a package called SOUNDPAS.ZIP on his Xevious board (508-875-3618) that generates sound in the background. It uses the system clock and can also be used to synchronize video frame updates. It is in Pascal. He may also have a C version I translated from some of his code and sent him (SOUNDC.ZIP). Also look in places like the libraries on Borland's CIS forum or other BBS's like Dan Linton's Software Creations (508-365-2359).

FAIR WARNING: If you fool with the system interrupts or directly modify video hardware, you should really consider Control-Break and Critical Error handlers. If your program crashes because a disk drive door was open, your user would like the program to recover gracefully, not take the entire system down. Code for this is also available in some advanced texts and in some of the libraries mentioned above.

Changing interrupts may also make your code into what Microsoft calls "poorly behaved applications [sic]". That is their way of saying that their Windows environment cannot deal with it. Try running a "poorly behaved application" like Duke Nukem or Crystal Caves from Windows.

Finally, a few words about the last item on the list. I mentioned using two screens if you have a lot of graphics to move. The advantage of a second screen is that you can be painting one screen while you display the other. When you are done you toggle screens, show the new one, and rework the old one. If you have been confused so far, I would suggest you not try this yet. It is amazing how much more confusing it is to deal with two screens -- much more that twice the work. Start simply, and work your way up. My personal experience has been that there is no substitute for actually trying a few things. Go write some code. Good luck!

BITWISE DATA COMPRESSION

Garrett Krueger

Despite the fact that computers are now coming with 4Mb, 8Mb, and sometimes more RAM, we still want to use as little memory as conveniently possible. When programming under MS-DOS, it becomes even more important to keep memory usage low. What is more, hard drives that we thought were huge a year and a half ago are now far too small. Program code takes megabytes, and data sometimes takes twice that. We want to reclaim much of this space. At the same time, we do not want to sacrifice program performance. How can we do this simultaneously?

One could, of course, develop only small programs; but this is generally not in the best interest of the end user. One could also use only small data structures, but again, this is inconvenient to the end user since many of the user conveniences require large amounts of memory from a programming standpoint. This is where data compression comes in.

There are many data compression methods available. Some forms are what I call "after-the-fact" compression while others are "time-of-storage" compression. "After-the-fact" is usually used on stored files and generally has no effect on the quantity of RAM consumed during program execution. "Time-of-storage" is used during program execution and can significantly reduce RAM consumption during program execution. Before going into specifics, I would just like to mention a couple of "after-the-fact" methods. This article, however, will focus on compression at the time of storage.

Two common forms of "after-the-fact" compression relate to the elimination of redundant characters in files. The first calls for reducing the number of NULL characters in files (especially executable code). Primarily the number of sequential NULL characters is counted. The characters are removed, and in their place two characters are stored. The first stored character is usually a special one that signifies compression is about to follow. The second stored character is the number of NULLs that were removed. You can see that more than two NULLs in a row would result in compression.

The other variation of "after-the-fact" compression looks for any characters that are sequentially redundant. To store this type of compression requires that three characters replace the sequence of redundant characters. As in NULL compression, the first stored character alerts the compression routine that a compressed character is about to follow. The next two stored characters are 1) a single example of the redundant character and 2) the number of redundancies eliminated.

While these methods can help reduce disk space used, they do not help to minimize RAM usage. I wanted to mention them, however, since they can be useful in the proper situation. As a more powerful alternative, "time-of-storage" compression does help decrease RAM usage during program execution and can also be used to reduce storage space required on disks.

This type of compression can be made specific to the information being stored. Because of this aspect, it can use the smallest possible storage option the computer has to offer -- the bit. Because working with bits offers a choice range of OFF or ON, encoding and decoding routines need not become too complex. The best conveyance of these methods is by example, so here we go.

Let us say that we wish to design a program to keep test scores for students. There will be

five tests during the quarter. In standard programming we might create a string or array with five spaces to hold the five character values representing the five test grades (one letter grade, A, B, C, D, or F, for each of the five tests). Since a single character requires one byte, this would take five bytes of memory for each student.

Can we compact this usage? Sure we can. Instead let us create numerical constants to represent each of the five letter grades: A_Grade = 1; B_Grade = 2, and so on. To store the grades, we will trade in the character string (five bytes) for a word-sized variable (two bytes). Now we can store each grade using only 3 bits of the word variable. This is because three bits can store a numerical value as high as 4+2+1 = 7. (In this case, we need only store as high as 5 which represents "F".) The resulting data structure appears something like that shown in the table at the end of this article.

If you have 100 students, you can save yourself 300 bytes of space just in this instance. Over four quarters this becomes 1.2 Kbytes. In terms of pure numbers, you have cut your storage needs by sixty percent. It can clearly be seen that this represents a significant savings on a large scale project.

Encoding and decoding for storage takes a little skill, and I will leave the details to you. In pseudocode, however, storage would probably look something like this:

```
EncodingRoutine
  Test 1:   word = word + x_Grade;
  Test 2:   word = word + (x_Grade        shift_left 3);
  Test 3:   word = word + (x_Grade        shift_left 6);
  Test 4:   word = word + (x_Grade        shift_left 9);
  Test 5:   word = word + (x_Grade        shift_left 12);

Main Module
For FirstStudent to LastStudent
      ReadInTestScoreOne   (score two, three, four, or five) and
            CallEncodingRoutine
```

where word represents the word storage value for the particular student, and x_Grade represents that student's grade on that particular test.

If you are astute, you may be saying, "There's an extra bit at the end that isn't being used." Well, if you want to, you can continue on from that point -- bit 15 -- to start storing Test 1 for the next student. Every three words you would have gained space for one extra test grade. This accounts for an entire additional student every 15 words -- a gain of 6.6 percent. Since storing this way would be even more complex, you may consider that the benefits would not outweigh the headaches to gain six more bytes.

Now, let us try another example. Let us say that you are designing a quiz of eight true/false questions. If you use a standard boolean variable or some other byte-sized representation to hold the answer for each question, it can cost lots of space (8 bytes per test). Here is a way to condense it. Use a single byte to hold all of the answers. Each bit in the byte will represent the boolean answer for one of the questions (bit 0 = question 1, bit 1 = question 2, and so on). You have now reduced required storage space by 87.5 percent. In the case of 100 students' tests, you can save 700 bytes. Again, over four quarters you have saved 2.8 Kbytes of storage space both on disk and in RAM while the program is running.

As in the previous example, encoding and decoding is similar. You would probably want to create two numerical constants to represent the options: TRUE = 1; FALSE = 0. Then, in pseudocode, your storage routine would look something like:

```
Question 1:   byte = byte + answer;
```

```
Question 2:   byte = byte + (answer shift_left 1);
Question 3:   byte = byte + (answer shift_left 2);
and so on...
```

where byte represents the byte storage value for your quiz answers, and answer represents the individual TRUE or FALSE value for that question. You can probably already see other ways to code the above. Again, I will leave the details to you.

Of course, the main thing to keep in mind is that these two examples represent "ideas" on how data can be compacted. You can experiment and come up with your own specific methods if you have not already.

I wish you good luck, and have fun...

Table

```
Student: |<                Student 1                >|    |<        Student 2...
Word:      |<                 Word 1                 >|    |<          Word 2...
Bit:   0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15     16 17 18 19 20 21 22 23 24...
Constant:   3       2       1        2          1              2          3
Grade:       C       B       A        B          A              B          C
Test:    |<1>| |<2>| |<3>| |< 4 >| |<   5 >|         |<   1 >| |<   2 >| |<   3 >|
```


- Main Menu

ADDING MOUSE SUPPORT TO YOUR PROGRAM

Gary Alston

I've seen problems with various mouse drivers. In general, to avoid most problems I tend to stick to working with only a small number of the mouse interrupts. As a general rule, you should verify that a mouse exists before attempting to use it. The following code allows you to verify that a mouse exists and provides other routines to allow you to implement mouse support. This code was written for PowerBASIC but can be easily converted to your language.

```
' assume global integer variable called mouse%
' that is used to indicate presence.

' use as:   mouse% = MouseInstalled

' also assume global integer variables MX% and
' MY% to record the last known mouse position.

function MouseInstalled
      MouseInstalled = 0         ' Assume no mouse
      reg 1, &h0000
      call interrupt &h33        ' try to initialize
      if reg(1) < 0 then MouseInstalled = 1
end function

sub ShowMouse
      shared mouse%
     if(mouse%) then
            reg 1, &h0001      ' turn on the cursor
            call interrupt &h33
      end if
end sub

sub HideMouse
      shared mouse%
     if(mouse%) then
            reg 1, &h0002      ' turn off the cursor
            call interrupt &h33
      end if
end sub

function MouseClicked
      reg 1, &h0003
      call interrupt &h33
      if reg(2) and &h0001 then
            ' this is left button, the others follow
            ' the button ID of 1-4-2 from left to
            ' right.
            MouseClicked = 1
      else
            MouseClicked = 0
      end if
end function

sub MousePosition
      shared MX%,MY%
      reg 1, &h0003
      call interrupt &h33
      MX% = (reg(4) / 8) + 1      ' this adjusts the
            'location for text mode
```

```
        MY% = (reg(3) / 8) + 1
end sub

sub MouseWait
        while(MouseClicked)          ' loop while the
                ' user's finger is still on the button.
        wend
end sub

Here's how I usually implement the calls:
'
'
local h%

if(not keypress) then
        if(MouseClicked) then
                MousePosition
                h% = HitTest(feed coordinates here:_                    xlow, xhi, ylow, yhi)
                if(h%) then...     ' process
                        HideMouse
                        ' do whatever!
                        ShowMouse
                end if
                ' next hit test goes here
        end if ' MouseClicked
end if ' keypress
Function HitTest(xl, xh, yl, yh)
        HitTest = 0
        if(MX% >= xl and MX% < xh and_ MY% >= yl and MY% < yh) then_ HitTest = 1
End Function
```


- Processing Interrupts
- Main Menu

SIMPLE USAGE OF EXTENDED MEMORY

Garrett Krueger

Recently I wrote an article on increasing the efficiency of DOS memory usage by employing bit-wise data compression at the time of storage. Now I would like to mention another effective way to use DOS's minimal 640K allotment -- don't use it!!!

"Ha, ha!" you laugh. "I have to use DOS memory!"

Well, okay, you can use some of it. The alternative I am getting at is to use Microsoft's eXtended Memory Manager. If you are programming software for 32 bit machines, this option opens up huge new memory frontiers for you -- 4 gigabytes. (Of course, now you become more limited by your pocket book than by the operating system.) Even 16-bit machines still give you 16 megabytes to work with.

By now, I am sure this is nothing new to you. The eXtended Memory System has been around for several years. Still, however, its full potential is not being used. One could store entire, large databases in extended memory to eliminate slow disk access. Temporary swap files can be stored there. If your program is large, you could run the entire program in extended memory using a DOS window as small as 32K or 64K. Dozens of graphic bitmaps for educational games/screens can be stored there. The list could go on....

Enough talk! Let us get down to the meat of the article -- some program code. The following program is one I wrote to simply and briefly explore the extended memory system on your computer. It was written to compile under Borland's Turbo Assembler (TASM), and I know for certain that it works with versions 1.0 and 2.5.

I believe the general code should work with Microsoft's assembler (MASM), however, the code headers (.MODEL, .STACK, .DATA, .CODE) may need to be revised. If you have MASM, you will probably know what to do.

I have tried to follow a logical progression in program design: check for extended memory and size available, allocate a portion for use with our program, store something in it, retrieve what was stored, and finally de-allocate the portion we own.

```
;-------------------------------------------------------------------------------
;   The following program compiles with TASM v1.0 and v2.5.
;   Program:   TESTXMS.ASM
;   Copyright (C) 1992 CleoSci All Rights Reserved
;
          .MODEL    compact
          .STACK    200h
          .DATA
errmsg1   db         'No XMS installed$'
errmsg2   db         'Not enough XMS$'
errmsg3   db         'Allocate Failure$'
errmsg4   db         'Free XMS Failure$'
errmsg5   db         'Error copying block$'
msg1      db         'Copying data to XMS...   $'
msg2      db         'Done!$'
msg3      db         'Copying data from XMS...$'
msg4      db         'Data in XMS was      : $'
OStr1     db        'This program checks for, allocates, and then de-allocates',0Dh,0Ah,'$'
OStr2     db        'XMS memory on your system.   If it is present, it allocates a',0Dh,0Ah,'$'
OStr3     db        '64K block.   If it is not available, the program terminates.',0Dh,0Ah,'$'
OStr4     db        'Error messages are displayed if appropriate.',0Dh,0Ah,0Dh,0Ah,0Dh,0Ah,'$'
```

```
BlkRequested   equ         64                          ;size of XMM request block
XmmPtr         dd          0                          ;pointer to XMM entry point
TotalFree      dw          0                          ;total amount of free XMM
LargestBlk     dw          0                          ;largest single free block
XMShandle      dw          0                          ;allocation handle
alloc          dw          0                          ;block allocation flag
errval         db          0                          ;holds errors returned

TotStr         db          'Total XMS Available','$'
LgStr          db          'Largest Free Block ','$'
InProc1        db          'Allocating             $'
InProc2        db          'Freeing                $'
AllocStr       db          'Allocated (shows 1)','$'
ErrStr         db          'Error code:            ','$'
CFLF           db          0Dh,0Ah,'$'
Result         db          ': ',4 DUP (?)
CountInsertEnd     LABEL BYTE
               db          ' KBytes',0Dh,0Ah,'$'

ParamBlk       equ         $                          ;XMM parameter block header
BlkLength      dd          0                          ;block size to move (bytes)
SrcHandle      dw          0                          ;handle for source block
SrcOffset      dd          0                          ;pointer to source block
DestHandle     dw          0                          ;handle for destination block
DestOffset     dd          0                          ;pointer to destination block
DataBuffer     db          'Hello, my name is Garrett$'
BufferSize     equ         $-DataBuffer
NewPhrase      db          256 dup (?)

               .CODE
ProgramStart:

               mov         ax,@data
               mov         ds,ax                      ;sets ds to point to data segment
                                                      ;
                                                      ; begin XMM system calling
               call        Opener
;
               call        SeekXMM
               call        GetXMMEntry
               call        GetXMMSize
;
               call        AllocBlock
;
               mov         bx,OFFSET TotStr
               mov         ax,TotalFree               ;Num to convert
               call        OutputValue                ;Total XMS
;
               mov         bx,OFFSET LgStr
               mov         ax,LargestBlk              ;Num to convert
               call        OutputValue                ;Largest XMS block
;
               mov         bx,OFFSET InProc1
               mov         ax,BlkRequested            ;Num to convert
               call        OutputValue                ;Allocate pass/fail
;
               mov         bx,OFFSET AllocStr
               mov         ax,alloc                   ;Num to convert
               call        OutputValue                ;Allocate pass/fail
;
               mov         bx,OFFSET ErrStr
               mov         ax,0
               call        OutputValue                ;Error number (if any)
;
;
               ; Now recheck mem avail after allocation
               mov         bx,OFFSET CFLF
               call        PrintStr
```

```
        mov     ah,8
        call    XmmPtr
        mov     TotalFree,dx
        mov     LargestBlk,ax
        mov     bx,OFFSET TotStr
        mov     ax,TotalFree            ;Num to convert
        call    OutputValue             ;Total XMS
;
        mov     bx,OFFSET LgStr
        mov     ax,LargestBlk           ;Num to convert
        call    OutputValue             ;Largest XMS block
;
;
        ;Copy data to XMS and try to read it back again
        ;into a different buffer.   Print the new phrase.
        call    CopyBlockToXMS
        call    CopyBlockFromXMS
        mov     bx,OFFSET msg4
        call    PrintStr
        mov     bx,OFFSET NewPhrase
        call    PrintStr
        mov     bx,OFFSET CFLF
        call    PrintStr
        mov   bx,OFFSET CFLF
        call  PrintStr
;
        ; Now free block and recheck mem after freeing.
        mov     bx,OFFSET InProc2
        mov     ax,BlkRequested         ;Num to convert
        call    OutputValue             ;Allocate pass/fail
        call    FreeBlock
;
        mov     bx,OFFSET CFLF
        call    PrintStr
        mov     ah,8
        call    XmmPtr
        mov     TotalFree,dx
        mov     LargestBlk,ax
        mov     bx,OFFSET TotStr
        mov     ax,TotalFree            ;Num to convert
        call    OutputValue             ;Total XMS
;
        mov     bx,OFFSET LgStr
        mov     ax,LargestBlk           ;Num to convert
        call    OutputValue             ;Largest XMS block
;
;
;
        mov     ah,4Ch
        int     21h
;
;
AllocBlock PROC  NEAR
        mov     ah,9
        mov     dx,BlkRequested
        call    XmmPtr
        or      ax,ax
        jz      error3
        mov     XMShandle,dx
        mov     alloc,ax
        mov     errval,bl
        ret
AllocBlock ENDP
;
Error3  PROC    NEAR
        push    ax
        push    dx
        mov     dx,OFFSET errmsg3
        mov     ah,9
```

```
        int     21h
        pop     dx
        pop     ax
        mov     ah,4Ch
        int     21h
Error3  ENDP
;
Error1  PROC    NEAR
        push    ax
        push    dx
        mov     dx,OFFSET errmsg1
        mov     ah,9
        int     21h
        pop     dx
        pop     ax
        mov     ah,4Ch
        int     21h
Error1  ENDP
;
FreeBlock PROC  NEAR
        mov     ah,0Ah
        mov     dx,XMShandle
        call    XmmPtr
        or      ax,ax
        jz      error4
        ret
FreeBlock ENDP
;
Error4  PROC    NEAR
        push    ax
        push    dx
        mov     dx,OFFSET errmsg4
        mov     ah,9
        int     21h
        pop     dx
        pop     ax
        mov     ah,4Ch
        int     21h
Error4  ENDP
;
SeekXMM PROC    NEAR
        mov     ax,4300h
        int     2Fh
        cmp     al,80h
        jne     Error1
        ret
SeekXMM ENDP
;
GetXMMEntry PROC NEAR
        mov     ax,4310h
        int     2Fh
        mov     word ptr XmmPtr,bx
        mov     word ptr XmmPtr+2,es
        ret
GetXMMEntry ENDP
;
GetXMMSize PROC NEAR
        mov     ah,8
        call    XmmPtr
        mov     TotalFree,dx
        mov     LargestBlk,ax
        cmp     dx,BlkRequested
        jb      error2
        ret
GetXMMSize ENDP
;
Error2  PROC    NEAR
        push    ax
        push    dx
```

```
        mov     dx,OFFSET errmsg2
        mov     ah,9
        int     21h
        pop     dx
        pop     ax
        mov     ah,4Ch
        int     21h
Error2  ENDP
;
Opener  PROC    NEAR
        mov     bx,OFFSET OStr1
        call    PrintStr
        mov     bx,OFFSET OStr2
        call    PrintStr
        mov     bx,OFFSET OStr3
        call    PrintStr
        mov     bx,OFFSET OStr4
        call    PrintStr
        ret
Opener  ENDP
;
CopyBlockToXMS  PROC    NEAR
                mov     SrcHandle,0
                mov     word ptr SrcOffset,offset DataBuffer
                mov     word ptr SrcOffset+2,seg DataBuffer
                mov     ax,XMShandle
                mov     DestHandle,ax
                mov     word ptr DestOffset,0
                mov     word ptr DestOffset+2,0
                mov     word ptr BlkLength,BufferSize;
                mov     word ptr BlkLength+2,0
;
                mov     bx,OFFSET CFLF
                call    PrintStr
                mov     bx,OFFSET msg1
                call    PrintStr
;
                mov     ah,0Bh
                mov     si,offset ParamBlk
                call    XmmPtr
                or      ax,ax
                jz      error5
;
                mov     bx,OFFSET msg2
                call    PrintStr
                mov     bx,OFFSET CFLF
                call    PrintStr
                ret
CopyBlockToXMS  ENDP
;
Error5  PROC    NEAR
        push    ax
        push    dx
        mov     dx,OFFSET errmsg5
        mov     ah,9
        int     21h
        call    FreeBlock
        pop     dx
        pop     ax
        mov     ah,4Ch
        int     21h
Error5  ENDP
;
CopyBlockFromXMS PROC   NEAR
                 mov    DestHandle,0
                 mov    word ptr DestOffset,offset NewPhrase
                 mov    word ptr DestOffset+2,seg NewPhrase
                 mov    ax,XMShandle
                 mov    SrcHandle,ax
```

```
                    mov    word ptr SrcOffset,0
                    mov    word ptr SrcOffset+2,0
                    mov    word ptr BlkLength,BufferSize;
                    mov    word ptr BlkLength+2,0
;
                    mov    bx,OFFSET msg3
                    call   PrintStr
;
                    mov    ah,0Bh
                    mov    si,offset ParamBlk
                    call   XmmPtr
                    or      ax,ax
                    jz      error5
;
                    mov    bx,OFFSET msg2
                    call   PrintStr
                    mov    bx,OFFSET CFLF
                    call   PrintStr
                    ret
CopyBlockFromXMS ENDP
;
;-------------------------------------------------------------------------
; Subroutine to convert a binary number to a text string.
;
; Call with:   AX      = Number to Convert
;                DS:BX = pointer to end of string to store text in
;                CX      = Number of digits to convert
;
; Returns   :   Nothing
; Registers destroyed : AX, BX, CX, DX, SI
;
ConvNumToStr   PROC NEAR
          push   ax                          ;preserve all registers
          push   bx
          push   cx
          push   dx
          push   si

    ConvertLoop:
          mov    si,10                         ;used to divide by 10 ConvertLoop
          sub    dx,dx                         ;convert AX to doubleword in DX:AX
          div    si                           ;divide number by 10.   Remainder
                                              ;is in DX--this is a 1 digit
                                              ;decimal number.   Num/10 is in AX.
          add    dl,'0'                       ;Convert remainder to a text char.
          mov    [bx],dl                       ;put this digit in the string.
          dec    bx                           ;point to location for the next
                                              ;most significant digit.
          loop   ConvertLoop                     ;do next digit, if any.

          pop    si                            ;return all registers
          pop    dx
          pop    cx
          pop    bx
          pop    ax
          ret
ConvNumToStr   ENDP
;-------------------------------------------------------------------------
;
;-------------------------------------------------------------------------
; Subroutine to print a text string.
;
; Call with:   BX = string to print (i.e. mov   bx,OFFSET StrToPrnt)
;
; Returns   :   Nothing
; Registers Destroyed : AX, DX
;
PrintStr        PROC    NEAR
          push   ax                  ;save registers
```

```
        push   dx
;
        mov    ah,9                ;dos printstring function
        mov    dx,bx               ;string to print (in bx)
        int    21h                 ;go ahead, print it
;
        pop    dx                  ;restore registers
        pop    ax
        ret
PrintStr        ENDP
;--------------------------------------------------------------------------
;
;--------------------------------------------------------------------------
; Subroutine to output a numerical value as a text string.
;
; Call with:   AX      = Number to Convert
;
; Returns   :   Nothing
; Registers destroyed : AX, BX, CX
;
OutputValue    PROC    NEAR
        call       PrintStr
        mov        bx,OFFSET CountInsertEnd-1    ;locate end of storage area
        mov        cx,4                          ;number of digits to convert
        call       ConvNumToStr
        mov        bx,OFFSET Result
        call       PrintStr
        ret
OutputValue    ENDP
;--------------------------------------------------------------------------
;
;
        END        ProgramStart
```

If you have extended memory available on your system and the program was copied, assembled, and linked correctly, your screen should show you something like this:

```
------------------------------------------------------------
This program checks for, allocates, and then de-allocates
XMS memory on your system.  If it is present, it allocates a
64K block.  If it is not available, the program terminates.
Error messages are displayed if appropriate.


Total XMS Available: 3008 KBytes
Largest Free Block : 3008 KBytes
Allocating         : 0064 KBytes
Allocated (shows 1): 0001 KBytes
Error code:        : 0000 KBytes
Total XMS Available: 2944 KBytes
Largest Free Block : 2944 KBytes

Copying data to XMS...  Done!
Copying data from XMS...Done!
Data in XMS was    : Hello, my name is Garrett

Freeing            : 0064 KBytes

Total XMS Available: 3008 KBytes
Largest Free Block : 3008 KBytes
------------------------------------------------------------
```

I would like to close by offering sources for more information. Chapter 3 (Ray Duncan) of the book Extending DOS details your alternatives when using the extended memory manager. You may also write Microsoft at: Microsoft Corporation, Box 97017, Redmond, WA 98073. Ask for the free publication Extended Memory Specification Version 2.0.

- Bit-Wise Data Compression
- Main Menu

PROCESSING INTERRUPTS IN C

Gary Alston

In the last ESC newsletter (Vol 1, No. 4), John Gallant had some good video animation tips, but said he'd leave the subject of interrupts to others. Interrupts can certainly be manipulated in C, and the following code will show you how it's done. The good news is that it does NOT require an assembler. The bad news is that you still have to know what you are doing....

We have been using this basic framework for writing TSR code for years now, and it works fast enough that we'll present it to you. Hopefully, some of you may be able to combine the techniques shown here with Mr. Gallant's general dissertation.

The function of type INTERRUPT is a special case with the Borland compilers, and I recall that MSC also supports them, albeit using a slightly different call. Essentially, type INTERRUPT is called whenever an interrupt is invoked. The first step, in function `main()`, is to use `getvect()` and `setvect()` calls. These are little more than a more convenient way to call the exact same function using DOS. `Getvect()`, when called, will point to the interrupt vector (i.e. physical location pointer). In this example, we're going to replace the 18.2 / sec timer tick interrupt at hex 1C. We will replace it with another INTERRUPT function that we'll call `OurInt`.

This is all really very simple: All we need to do is to use `getvect()` to read the current interrupt vector, and use `setvect()` to point to our `OurInt` function which will replace it. Note that `setvect()` will essentially replace the old interrupt vector with the (dynamic) physical address of the `OurInt` function.

The functioning of `OurInt` is also pretty straightforward: `sflag` is used to tell us if we are already in process in case we get interrupted in our own process. `OurInt` also uses pseudovariables to record the register contents and set up a stack. The size of this stack is going to depend on what you are doing. Note that you'll generally want to keep everything the interrupt takes care of pretty small to start with.

The code that you write inside the `OurInt` function depends on what you want it to do. What you CANNOT EVER do is use any function that uses a DOS call. Writes to ports, using BIOS, direct memory manipulation and so on are fair game. Sometimes you may want to use a C library function. If you are unsure if it calls DOS, simply compile the program in command line mode to have it output the assembler code. In Borland implementations, it looks like:

```
BCC [options] -S (asm listing) [filename]
```

This general purpose routine is very handy. We have used it as the shell for putting clocks on-screen, reading touch pads and fooling a program so it thinks it's seeing keyboard keystrokes (by inserting key and scan codes into the keyboard buffer) and so on. There's no reason why you can't use this to do memory moves.

Restoring this when you are done is simple, too: just reverse the calls.

In summary, I hope that if you were unsure of how to access and revector system interrupts that this will help. I can also recommend the book The Art of C -- Elegant Programming Solutions by Herb Schildt, published by Osborne / McGraw-Hill (ISBN 0-07-881691-2). It has a

very good section on this subject and contains working sample code on disk.

See following source code examples.

```
#include <dos.h>
#include <stdlib.h>
#include <process.h>

void interrupt (*oldclock)(void);

// GLOBAL VARIABLES
int savess, savesp, sflag;
char stack [0x1000];

void IntSetup(void)
{
   oldclock = getvect(0x1c); // clock timer
   setvect (0x1c,OurInt);      // re-vector
   keep (0, 2000);             // set aside 32k and KEEP dos from reallocating
}

void interrupt OurInt (void)
{
      if(!sflag){
      savesp = _SP; // save the current stack pointer
      savess = _SS; // and stack segment
      _CX = (int)&stack[sizeof(stack)];
      _SS = _DS;
      _SP = _CX;
      }
      sflag++;

      // YOUR CODE GOES HERE

if (!--sflag) {
      _SS = savess; // restore the stack pointer
      _SP = savesp; // and the stack segment
     }
}
```

IS SHAREWARE DEAD?

Randy MacLean

For over twelve years, I've participated in the vibrant and radical world of the shareware community. During that time, I've seen fortunes made and lost, conventions turned upside down, and the coming (and sometimes, sadly, the passing) of the greats of our business.

Recent trends have made me wonder if the shareware phenomenon has passed its peak. Although the evidence is more anecdotal than scientific, it's still cause for pause.

Item: The once-burgeoning ranks of disk vendors have experienced a precipitous drop. Many of the key vendors of yesteryear have disappeared, and overall the number of active vendors is down (by some reports) to less than half of their peak levels.

Item: The number of new authors entering the business is below what we experienced only a year or two ago. Meanwhile, many of the best known shareware operations are deriving a growing proportion of their business from more traditional activities -- contract consulting and retail publishing.

Item: Many authors are reporting lower registration earnings. Although there are those with focused marketing efforts who are expanding their activities (and their income), many of the more casual operations are experiencing a general decline in revenue. A lot of authors are finding it increasingly difficult to maintain peak sales levels.

All this is in sharp contrast to the two bright spots in the industry: The explosion of BBS's and the new-found celebrity of the Internet are bringing new users to the online community in droves. Modem sales are way up, and there are more users of online services than ever before.

Traditional software publishers are embracing shareware distribution as never before. Players like Interplay and Lucas Arts have discovered the power of shareware distribution, (advertising that pays, not costs). After some reflection, I've come to the conclusion that these changes are a natural consequence of changes in our users' environment.

All of us can remember back to when computer software outlets were rare indeed. In those days, the shareware marketing system (and mail-order vendors) gave users a broader selection of software in a relatively convenient home delivery system.

Today, software superstores are within driving distance of 80% of the population, delivering an incredible range of choice at competitive prices. The gaps are largely filled by a myriad of retailers, from video stores to the local Radio Shack. A user can drop into a nearby store and immediately pick up the perfect program for his needs. For the user, there's no need to wait for mail delivery of a great shareware offering.

The downward pressure on software prices has also eroded the price advantage shareware products used to enjoy. These days, almost any kind of package can be had for under fifty or sixty dollars.

So, where does this leave us? As usual, with a lot of opportunity! First, shareware retains its original advantage - users can evaluate our software before purchasing it. The trick is to make sure that they actually try it out before they head down to the store. Bundling with other programs, software collection CD's, and online access are more important than ever --

you've got to get your software into the hands of as many users as possible. Make sure the delivery is coherent -- the user has to be able to find your programs. There's no benefit if you're buried on a CD with a million ZIP files.

Second, we should still enjoy a price advantage. We're not saddled with skyrocketing marketing costs the retail guys face. Of course, this means we'll have to sell more copies to keep our revenue up, but we're in the software business and everyone in our business has to deal with that. In the meantime, users have a right to expect to pay less for a shareware program (with no fancy manual, packaging, etc).

Third, we've got excellent access to the retail marketplace. Retail publishing deals can be had by many authors, and nearly everyone can get onto a retail CD. Bundling deals are more popular than ever, with software and hardware vendors always looking for ways to add value and differentiate their offerings. This gives you the chance to be in the user's hands before he heads down to the store to purchase your commercial competition's offering.

Fourth, we can beat the retailers at the "instant gratification" game. Through the use of online registration services, ZIP unlocking and other systems, we can make obtaining our programs faster and more convenient than our retail competitors ever can. Any method that'll deliver the registered version easily and immediately will knock the retail publishers out of the running.

I expect that there'll continue to be casualties in the shareware community. Like the mail-order vendors who failed to make the transition to retail, those who continue to do business the old way -- the way that used to work -- will have an increasingly difficult time making shareware pay.

However, the brightest and most nimble -- those who understand and take advantage of the evolution of the market -- will continue to prosper, as they always have.

So is shareware dead? Not for those who can understand and adapt to the evolving environment. The future will be interesting to watch.

SHAREWARE MARKETING STRATEGY

Tim Sweeney

Shareware is a unique way of marketing software, and it needs to be approached with a good strategy to be profitable. The Epic MegaGames philosophy is clear, simple, and it works extremely well.

We make shareware successful by giving our customers an excellent, complete, and top-quality product for free -- then we sell them more of what they want: Even more excellent, complete, and top-quality products.

It is this simple strategy that is responsible for Epic's success -- this is why a good Epic game can bring in 20 orders per day, while most other shareware typically gets two orders per week. Most shareware fails to sell because most shareware authors don't know how to use positive marketing -- but we do, and therefore we are successful.

Here is how most authors fail: They do not make their potential customers happy. When a customer is happy, he will be in a good mood to order. An unhappy customer will not order. To make our potential customers happy, we give them great shareware games -- fully working, top-quality products that they can play and enjoy for free. This makes our customers happy.

Once a customer is happy, he is ready to buy. Our next step is to sell him something. We do this by offering more games, and extra bonuses: hint sheets, cheat codes, bonus game disks, and more. These extras are incentives for our customers to order.

Here is why other shareware authors fail:

1. They cripple their products -- for example, a game without a "save" feature -- you have to register if you want to save your game. This fails because customers end up being mad at the author; unhappy customers won't order.

2. They don't offer any incentives to customers. Amazingly, many shareware authors basically say this:

> "If you like this program, please register by sending $30. You will receive the latest version of the program and the author's gratitude."

But what are they selling? For $30, the customer will receive something he already has! That is not an incentive for ordering -- you need to sell them something they do not have, and would like to have.

Here is why Epic's games sell so well in shareware:

1. We give customers great, fully-working software free for them to enjoy, which puts them in a good mood -- they like us and they trust us, so they are glad to do business with us.

2. We sell them something they want to buy: More games, hint sheets, cheat codes, bonus games, and more. Customers love this stuff!

Trilogies!

Epic's secret to success: All of our games are trilogies -- a three-volume series of games. We give away part 1 as shareware, and we sell parts 2 and 3 to customers. Yes, we have to work harder to create three episodes of a game, but it pays off. Customers love the first episode of our games, and are happy to get the other episodes from us. It's a great deal!

Game Requirements

The best-selling games in shareware are all trilogies. That's why it is so important to design a game with a 3-part series in mind. Several kinds of games are very easy to adapt to become a multi-part series:

1. Arcade-adventure games like Jill of the Jungle: Many different levels and worlds.
2. Pure arcade games like Kiloblaster: Many different levels.
3. Role-playing: New scenarios, creatures, and objects can be added.
4. Strategy games: New scenarios and, creatures, and objects can be added.

Quality, Technology, and Gameplay

These three factors are the absolute most important pieces of a successful shareware game. Fortunes are made or lost based on quality, gameplay, and technology. Thousands of shareware games are released every year, but only a select few make it to the top. These are the absolute highest quality, best games, with the most modern technology possible.

Quality: Games these days need to be designed to be 100% bug- free, compatible, easy to operate, and extremely "clean" in terms of graphics, sound, and gameplay. When it comes time to finish a game for release, our authors are not just coders -- they are Software Engineers -- with the responsibility to totally test, debug, test, polish, test, and test their product. Customers expect only the best from the Epic MegaGames team, and we will work to deliver. It's not easy, but this is our job.

Technology: Dazzling graphics, dazzling music, dazzling sound effects, dazzling techniques. These are the common factors linking all shareware success stories. In terms of graphics, customers will come from around the world for smooth, 256-color, animated VGA graphics that exploit their CPU's to the limit. If you've seen a technique used before, it's not good enough -- we need to push the PC to the end of its limits, then push it some more. That is how we'll leave the competition behind.

Gameplay: No matter how much quality and technology a game has, it can only be successful if you have a great core game. Customers want fun and innovative action which will keep them entertained for many days of play -- with new twists of plot around every corner, new scenarios being revealed every step of the way, and vivid variety.

Summary

Quality, technology, and gameplay make our games successful -- and guarantee that Epic's shareware will spread throughout the world and generate an instant following. The competition is good, but we're better -- and we are working very hard to be the best.

Once we have a good product, it's our customers who ultimately make shareware profitable. By giving the best, top-quality games to them, we have the opportunity to sell them more great games and bonuses -- by keeping the customers happy, we'll be successful.

Our customers win, our authors win. This is the Epic MegaGames way.

Vol 1, No 7, February 1993

- Main Menu

THE WORLD OF MULTIMEDIA

Orlando Dare and Clarence S. Wright, Jr.

The dank, cold walls of stone in the underground bunker are punctuated only by paintings of "Der Fuhrer" and massive, steel doors. A steel door opens with a clang. The hero bursts through the doorway, a German Schmeizer machine-pistol in hand. Two German Shepherd police dogs bark furiously, then attack. Several quick bursts of fire from the machine-pistol, and the hounds yelp, then expire. The hero whirls to his left as he hears something. An SS Officer approaches, his footsteps clearly audible. He begins firing his Luger. Another quick burst from the hero's machine-pistol, and the SS officer dies, his final breath crying out "Ach, mein Lieben!"

This isn't the scenario of a movie or even of a kid's arcade game. This is a scene from Wolfenstein (tm), a multimedia game program that thousands of PC buffs are enjoying on their home (or office) PCs.

Just what is multimedia? It may be defined as the simultaneous presentation of a series of effects in more than one media or format. That may include audio (sound), high resolution graphics (video), animation or a combination of all three.

I'm certain that all of us can recall our high school days when the teacher would drag in a fairly massive roll-away with a 16mm sound projector. Someone would have to set up the screen. Then we would watch a movie presentation by the National Farm Bureau called "Soybeans Are Our Friends" or some such similar title. Then, if the teacher wanted to present an accompanying slide show, in came another roll-away with an overhead projector or perhaps a film-strip projector hooked to a record player. That was eventually replaced by a TV set hooked to a VCR -- a slight improvement. A/V or Audio-Visual systems were, for the most part, bulky because it took several different kinds of equipment to achieve the effects we wanted.

That's ancient history these days. The desktop computer with a few peripherals can do it all -- and much better, at that. A desktop computer can be equipped with a sound card, high resolution graphics card, external speaker hook-ups and either a CD-ROM drive or a hard-drive (or both) and you're ready to rock 'n' roll. Instead of a computer monitor, we can use full color liquid crystal projection plates that fit on an overhead projector for wall-sized presentations.

Multimedia represents a breakthrough in computer aided audio-visual technology. Imagine turning dull business meetings and boring school classes into show-stopping, eye-popping presentations, complete with CD-quality speech, music, special effects and high quality graphic images. We're not talking about something in the distant future. We're talking about today -- right now !

Unless you've been on some remote island in the Pacific, you've probably heard the term "multimedia" repeated as if it were some high-tech mantra. Perhaps, not knowing exactly what multimedia really is or how easily the average PC can be upgraded to make use of multimedia, you probably tuned it out. However, multimedia is an ever-growing force, making its foothold in businesses, schools, corporations and in the home, and is now edging its way into networked applications.

Multimedia has become an industry catch-word for specially equipped PCs capable of

delivering full motion video, stereo sound and near-photo quality still pictures. The technology isn't all that revolutionary, really. Just take any PC with plenty of speed and data storage space, add a sound board and stereo speakers and a special compact disk player capable of storing today's complex multimedia software programs and you've got multi-media. Some programs use laser-disk players that have been on the market for over a decade.

"Why," you ask, "should we bother with multimedia?" One reason is to get and hold the viewer's attention. Multimedia grabs and holds your attention. According to studies, audiences retain twenty percent of the information they hear, forty percent of what they see and sixty to seventy percent of the information learned through interaction. Multimedia, by appealing to more than a single sense, gets the viewers more interested and allows them to retain a larger percentage of a particular message.

Schools compete for the attention of kids raised on electronic TV games and rock video. The same kids are, for the moment, the most avid users of multimedia programs. On the adult level, multimedia presentations are a growing force in the marketplace. It is being used in sales presentations, as an information medium in the political arena and as spot presentations in stores and shopping malls to attract business. The 1.44 megabyte multimedia "demo disk" has become a commonplace means of attracting sales to new forms of PC programs. That same 1.44 megabyte disk can be used for a full color, sight-and-sound multimedia presentation lasting ten minutes or more. As such, it is being used to tell people about our national parks, to sell time-share condominiums and as an "electronic yellow pages" system to help people plan their vacations in interesting places.

"So," you ask, "what's all the big fuss over multimedia? It sounds nice, but..." The problem is that the average person has several basic misconceptions about the uses of multimedia. These misconceptions tend to color the way that they look at anything connected with multimedia.

Error #1: You have to have a "monster machine" PC in order to effectively make use of multimedia. WRONG! Any PC, 80286 or higher, with a good hard drive, can be easily upgraded to multimedia. You will need a good VGA graphics card (preferably with 1MB-RAM) and VGA monitor. For memory, it's best if you have one or more megabytes of RAM on your motherboard. Multimedia sound boards sell for about $100.00 and can be hooked into existing external sound systems (such as your home stereo.) Complete multimedia systems sell for about $800.00 to $6,200.00. If you want to access any of the numerous multimedia programs on CD-ROM, the drive units start at about $300.00 and up.

Error #2: Multimedia is mostly for kid's games and high-tech stuff. WRONG! There are numerous multimedia programs already available covering nearly every facet of the wide range of PC programs. Want an entire encyclopedia on one CD-ROM disk, complete with multimedia data about almost anything and everything? How about User-Interactive teaching and tutorial programs? Games (for children and adults)? Spreadsheets? Business Presentations? Multimedia covers it all and more. New multimedia applications are appearing daily.

Error #3: Creating your own multimedia program is far too expensive, takes too much time and requires a PhD in Computer Nerdology. WRONG, AGAIN! The average commercial multimedia development program starts at about $400.00. There are shareware multi-media development programs for under $100.00. Most programs are user-friendly and require a learning level equivalent to writing batch files at the DOS level.

Multimedia can take on the role of a trainer, teacher, guide, and presenter as used in training, education, sales and marketing applications. Interactive training or educational

programs allow the user to move at his or her own speed, to review sections of a program that are not fully understood, or to look more closely at topics of interest.

How complex can it be? Well, as an example, if a students are learning about Dr. Martin Luther King, Jr., they can supplement what they read by seeing and hearing Dr. King's "I have a Dream" speech in multimedia. Students can cut and paste the information into a multimedia school report on Dr. King, complete with text, narration and actual clips of Dr. King at work.

Multimedia is becoming increasingly more popular because it combines text, graphics, video, audio and animation on a PC to provide a powerful communications format. It has already invaded the area of electronic "hypertext" books and tutorials with User-Interactive "desktop publishing" books that include graphics, sound effects and CD-quality digitized sound. The technology is even beginning to infiltrate networks. There are several forces behind multi-media's increasing demand and popularity. Rapid technological advances make multimedia more affordable and practical. Major software developers have arrived at a working multimedia standard. In addition, vendors are creating alliances for multimedia software and hardware development.

Meanwhile, in the software arena, specialized programs, such as games, educational reference packages, authoring tools and animation are quickly making inroads into the mainstream. Numerous hardware developers are offering new desktop PCs equipped for full multimedia use and development for prices ranging from $1,500.00 to $3,500.00. As a result of this, software developers are including multimedia capabilities in otherwise staid, business programs. Even word processing, database and spreadsheet programs are sprouting multimedia features. This development is largely due to such technological revolutions as the QuickTime (tm) real-time synchronization of video, animation, graphics and sound. MWindows (tm), a version of Windows (tm) 3.0 with multimedia extensions, is also a prime moving force behind this development.

Recently, however, multimedia has come to mean something entirely different to game makers and thus to game players. The term has somehow evolved to become a synonym for compact disk-based entertainment, as if the storage medium, alone, determines the genre.

Actually, the change in emphasis shouldn't come as any great surprise. CDs hold enormous amounts of data that can be easily and quickly accessed. This acts as a magnet to developers who have been putting six, eight, ten or more floppy disks in their boxes. CDs must look like the seemingly limitless Steppes of Russia to developers, offering open byte-space that seems to stretch from horizon to horizon.

It's nothing new, really. With the development of Upper-Level Memory and 100+ megabyte hard drives proliferating the market, software developers create larger, ever more complex programs. In the "good old days" when 640K-RAM and dual floppy drives represented the top of the market, software was small, tightly written and somewhat simplistic. The idea was to get in, do the job and get out. Not today! DOS, which used to occupy a single disk in its working format, now takes up half a dozen disks in compressed form. OS/2? Get serious! Some programs are normally packaged on CD-ROM disks. Software developers look at the "wide open spaces" on a CD-ROM disk and, like land developers, start looking for ways to fill that space with features that will force games to grow up. You'll get more elaborate sound effects, album-like sound tracks, detailed graphics bit mapped or object-oriented backgrounds and more talk. Graphics in the 1024 X 768 SVGA mode will become more common, but will eventually be replaced by much higher graphic formats. Games publishers see multimedia CDs as the next logical step upward from games that originated on floppy disks. In some cases, the CD is simply a substitute delivery system that replaces what

would have been dozens of high density disks.

One of the better examples is the CD version of Secret Weapons of the Luftwaffe (tm), a World War II flight simulator that pits American planes against the futuristic Nazi jet and rocket-propelled aircraft. The CD not only holds the game, itself, but also contains several second-purchase packages which add planes to both sides' inventories. The only difference between the CD-ROM version of the program and the floppy-based version is the fact that the CD-ROM version contains the extra aviation "inventory" that purchasers of the floppy-based version will pay extra to get.

Let's take a brief look at some of the multimedia applications in the past and a peek around the corner into the future to see what we can expect.

Schools: In New York, a public junior high school is using multimedia along with traditional PC programs as a central part of its technology-based curriculum. "[Multimedia] lets me explore stuff and do what I have to do by myself," says Jacqueline Rivera, an eighth grade student at the School of the Future. Her recent multimedia presentation on Benjamin Franklin earned her a B+, she says, with a smile.

Business: Computerized flight simulators have long been a common training tool at American Airlines. Now, American is using multimedia to let other employees train themselves and to allow travel agents to provide customers with up-to-date video tours of distant resorts. Its multimedia program, which shows and tells American's chefs how to prepare in-flight meals, won a recent airline industry award for innovation.

Information: Arlington International Raceway uses an IBM multimedia package to educate patrons on the basics of horse racing and wagering. By touching a screen, visitors can retrieve footage of famous race horses, interviews with jockeys or a rundown of track side restaurants. One interesting twist: Hitting the sushi icon gives Japanese race fans flight information to Tokyo and directions to a famous sushi restaurant there.

From the modest beginning of the common PC comes the rich sound and multimedia interactivity of today's PC -- one integrated system that can play audio, show animation and display high-resolution video graphics. You can explore CD-ROM based travel brochures or mail-order catalogs, complete with the engaging sights and hi-fi sounds and full color animation. Create your own artwork and animate it. Link your multimedia PC to a keyboard to record, store and replay MIDI music. The possibilities are as varied as your own imagination.

The workplace has become a dynamic, multi-sensory environment. Decisions are made and actions are taken based upon how clearly a person's meanings, actions and ideas are presented and perceived. Using multimedia, you can enrich your presentations with sound, music and high quality images. Interactivity helps you to customize your communication to each audience's specific interests.

Need to train someone? You can both show and test trainees about product demonstrations, diagnostics or any other task. You can also use multimedia to control laser disk players, digital video boards or virtually any other type of media device. In standard productivity applications such as spreadsheets, databases, word processors or electronic mail (email) multimedia annotations help you to quickly convey what text alone cannot. Remember, a picture is said to be worth a thousand words. How many words is a picture, animated with sound, speech and music worth?

Let me ask you a few questions so we can see some of the ways that YOU can use multimedia software and hardware. For each, we'll pose a little preliminary scenario for

your consideration.

You are a grade-school or Sunday-school teacher. You know that getting the student interested in what you have to say is the first, and most important half of the battle. (The second half of the battle is trying to satisfy the demand for knowledge once that interest has been sparked.) You have a PC with multimedia software and hardware and a VGA graphics card and monitor.

Question: Wouldn't it be easier to write your own "presentation" for your class? (Using a simple hand-scanner to scan in images and coloring them using any of a number of easy-to-use "paint" programs makes it possible for you to review a student workbook with all of your students at once.) Question: Using the graphics you have already designed for the classroom presentation, couldn't it be easier still to create a "review test", putting copies of that test into each student's PC?

You are a salesman working for ABC Widget Corporation. You've just gotten word from a friend that one of the Corporate High Muckety-Mucks will be present at tomorrow's sales meeting. They are looking for a new Mid-West District Sales Manager. You have tomorrow's presentation all worked up on a flip folder (just like everyone else) with charts and graphs and sales projections.

Question: Wouldn't your sales presentation have a greater impact if you used the charts and graphs you have already prepared on your PC as part of a multimedia presentation? Question: How soon could you and your family be ready to move to Denver, Colorado as the new Vice President of Sales for the Midwest Region?

You are a local small business owner or manager. You have been asked to give a seminar to other SB Owners and Managers on Business Techniques for the 1990s by your local Chamber of Commerce. You could give your presentation or seminar using the same tired, old flip-charts and what-not that have been used for the past fifty years. The odds are pretty good that more than half of your audience will be sleeping peacefully before you get to the third chart. Question: If you create your own multimedia presentation (a) Won't your audience get more out of what you say? (b) Won't these other business owners and managers remember you and what your company represents when they have need of the products and services you have to offer?

Are you beginning to get the drift?

Now, you have been reading a magazine article. At best, the printed page is a two-dimensional media of information presentation. (My apologies to the editors and publishers of this magazine, but facts are facts.) There are literally thousands of ways that the information in this article could have been presented in multimedia that would have been more interesting and much more informative. Perhaps (or probably) the magazines of the future will be presented as Electronic Publishing with full hypertext and multimedia capabilities. We have the ability to do so right this minute. However, publishers that use the traditional paper media will continue to use the paper media only so long as the public does not demand (at the cash-register level) that we move into electronic publishing, hypertext and multimedia modes. Multimedia and its sister informational presentation mode, electronic publishing, represent the future of PCs in our society.

I'm reasonably certain that the owners and users of the horse and carriage thought that the horseless carriage was "just a passing fad," but it doesn't work out that way. Multimedia, like the "horseless carriage," is here to stay. It isn't merely a part of the future. In the information age, it is the future.

- Main Menu

PRODUCT DIARY

Richard Goulet

As a new author of a shareware product (Mr. Spell) I've just started what I consider one of my most important activities, keeping a product diary. Humans have medical records and financial records that track two different types of health; both deal with the past. The purpose of a product diary is to track the past, but more important, it serves to help in the area of planning future marketing. If you think of a software product as a living thing, then like a doctor you want to know a lot about it to keep it healthy. Since you have a lot more control over your own software product than a doctor, the plan should not just be to keep it healthy, but also to make it grow and prosper. There are three different types of information I will be keeping in my product diary: miscellaneous record keeping, financial information and distribution tracking.

Misc. record keeping:
- publishing date
- beta testing dates and information
- all version names and release dates
- in-house archival storage names for all versions
- name and date tracking of any contracts
- any PR (public relations) dates

Financial (monthly):
- registrations and upgrades
- income
- expenses
- profit

Distribution tracking:
- when and where each version was sent

The product diary is not meant to go into any kind of detail. Like a photo album, it collects a snapshot of the product's life. The record-keeping section is needed because I have a bad memory; it's the other two sections that are important to support future marketing. Financial information is a sign of health: is the product growing or dying? How long after filling the distribution channels does it take to see significant income? Does the time of year affect sales? How long does it take for a release to get old? How effective was any PR?

Distribution tracking is important to help explain financial results and to plan future marketing. Did getting to be on a cover disk make a big difference? How many sales can be tracked to online services, CD-ROMs, the different vendors and to the local area BBS? Is marketing through low cost retail better than shareware?

The purpose of tracking is both to understand how the shareware distribution channels are working for you, and more important, to give you the information you need to try to improve your marketing. For example, if you don't get the largest number of registrations from California, then this information shows you that you have both a problem and an opportunity.

With tracking you look for trends as well as anything strange. Can you spot the future from small but steady changes in your sales? Are CD-ROMs replacing vendors or BBSs as your

best source of sales? Why does one small vendor produce more sales for you than the larger vendors? Did getting a software agent in Australia increase sales? Why do you get more sales from Iceland than England?

Assuming that you have created a product that there is a demand for, and that you have included good positive reasons for people to register it, then there is only one step left, marketing it. You can let the shareware channels move your product around for you and hope for the best or you can study the distribution channels and actively try to improve your marketing. Watch your product's flow through the different channels as well as which channels produce the highest returns.

You can improve your marketing by first getting to know the patient and then by coming up with ideas of how to treat what ails its health. The product diary helps you to do this for each of your software creations.

FILE_ID.DIZ

Bruce Jackson

Here are a few of the problems I see when I am trying to post files on my BBS. The different complaints come from several SysOps. So, although I have compiled this listing, not all of this comes from me.

1. The programs are not archived. Please, when submitting, make sure you submit a ZIPped file (or files if your program's size requires you to break it down into two or more parts). ZIP is the extension created when compressing a program with PKZIP, an archiving program by PKWare available on most BBSs. If the program is not archived, the sysop must work harder, and this also allows the program to be circulated under many names. PKWare's ZIP is the most widely used archive program. Sysops who are not set up to use ZIP will be set up to change from ZIP to the archiver of their choice.

NOTE: Many programmers submit files that do not have incrementing version numbers in the filename. This can be done if the only BBSs the program is on are the ones it is shipped to, but the distribution is cut way back. This is a biggie on the ESC CD-ROM as I have ended up killing somewhere between 30%-50% of the programs because naming conventions were not followed, causing me to put older versions back on line. In some cases this was not discovered until the newer version had been killed. Please, find the convention that works for you and stick with it. I can find the old program, delete it, and then run several batch files to make the various lists used by my utilities to decide if the upload should be allowed. However, I cannot upload XYZ.ZIP to a BBS that has it already, no matter which one is newer. A better way would be to release XYZ1.ZIP so when you release an upgrade, XYZ2.ZIP can be uploaded without the hassle to any BBS while allowing you better control of the distribution of your product.

Many sysops rename programs submitted without version numbers in the file name name, or just do not post them. This limits the effectiveness of your distribution and allows older, less feature-filled versions, to stay in the marketplace long after they should have been deleted. Either way, placing the version number in the file name just makes good sense. Note that in some special instances such as CompuServe and ZiffNet, it is requested you do not use version numbers, but these are exceptions.

2. The programs are hidden in subdirectories on the disk. Once again, please, make it easy for the sysop to get to the file. Place all submissions in the ROOT directory of the floppy. If you have multiple products, put each one in a separate ZIP in the root directory. NOTE: When processing files, many sysops just "COPY A:*.ZIP" and if your program does not copy, you do not get posted on that BBS. Please, for maximum distribution, keep your distribution archives in the root directory of your distribution floppy.

3. The programs sometimes come in a self extracting EXE file and not in a single ZIP. The self-extracting archive adds to the size of a file and allows another possible location for a virus to reside and so is rejected by many BBSs because of this one factor alone.

4. Submissions do not contain a FILE_ID.DIZ or have one that is "improper". This file is a small text file within the archive that describes the program. It must be no larger than 45 characters per line with a 10 line maximum. It must contain only text and no extended ASCII.

The "improper" FILE_ID.DIZs have contained blank lines and extended ASCII, exceeded 45 characters per line or 10 lines per file, or were not named FILE_ID.DIZ.

The file named FILE_ID.DIZ contains a description that, in its simplest form, has a maximum of 10 lines with a maximum of 45 characters per line. Here is the format:

```
x xx xxx xxxx xxxxxx xx xxx xxxx xxxxx xx xxx
xxxxx xxxxx xxx xx xxxxxx xxxx xxx xx x xx xx
x xxx xxx xxxx xxxxx x xx xxx xxxx xxxxx x xx
xxxxx xxxx xxx xxx xxxxx xxxx xxx xx x xx xxx
x xx xxx xxxx xxxxx x xx xxx xxxx xxxxx x xxx
xxxxx xxxx xxx xx xx xxxxx xxxx xxx xx x xxxx
x xx xxx xxxx xxxxx x xx xxx xxxx xxxxx x xxx
xxxxx xxxx xxx xx x xxxxx xxxx xxx xx x xx xx
x xx xxx xxxx xxxx x xx xxx xxxx xxxxx xxxxxx
xxxxx xxxx xxx xx x xxxxx xxxx xxx xx x xx xx
```

Notice:

1. No control characters or extended ASCII.
2. No blank lines.
3. No center justification. (Is left justified)
4. No bullet format.
5. No leading spaces.
6. Must be called FILE_ID.DIZ

This is a poorly-constructed DIZ file:

```
EIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII"
o Washington, DC area BBS list   Feb 1, 1994 o
o 577 -   BBS numbers verified monthly          o
o          17 down - 24 new local #s             o
o   16 -   Non-Metro                             o
o                                                o
o Non-metro sysops need to validate their     o
o listing or be deleted. See SYSOP.DOC file.o
EIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
verification by Mike Focke              2/1/94
```

1. Extended ASCII
2. Contains information that says nothing of the file.
3. Large quantities of wasted space.

Here's the 80-column format if the upload processor assumed the DIZ was correct:

```
DCBB0294.ZIP    118807   01-31-94 xEIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII" o
Washington, DC area BBS list   Feb 1, 1994 o o 577 -   BBS numbers verified
monthly          o o          17 down - 24 new local #s             o o   16 -   Non-Metro
o o                                                  o o Non-metro sysops need to
validate their     o o listing or be deleted. See SYSOP.DOC   file.o EIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
verification by   Mike Focke
   2/1/94
```

Here is a well-constructed DIZ file:

```
PERSONAL CALENDAR v14.62 <ASP> - TSR PIM.
```

Contains a printable appointment reminder (tickler), clock, scrollable calendar, notepad, and historical track. Deluxe, flexible, friendly Personal Information Manager, runs as normal program or stable, environment sensitive 6K byte TSR. -AV by Author, Paul Munoz-Colman. From FunStuff Software $35. DOS 6 and OS/2 compatible!

1. The most critical information in first line, next most critical by line 2, rest of critical information by line 4.
2. It's an informative paragraph, not bullet format.
3. It's NOT ABUSIVE TO THE EYES WITH ALL CAPS except for the program name and version, where this is strongly recommended!
4. It tells who it's from and the price.
5. It's not hard to reformat to 80 column format.
6. The ASP is in angle brackets. (More users search for this string than any other when seeking ASP files. I recommend ESC use a similar default.)

Here is the basic format for programs that require more than one ZIP file:

First DIZ:

Periodic Table 2.02 <ASP> Win3.1 (1of2)
The ultimate periodic table. It shows all 110 chemical elements. Gives over 30 data items for each one. View element details on multiple elements via its MDI interface. It has data on over 800 isotopes and can display radioactive decay trees for over 500. It has a quiz mode to test your knowledge.

Second DIZ:

Periodic Table 2.02 <ASP> Win3.1 (2of2)
The ultimate periodic table. See disk 1 for details.

or

Periodic Table 2.02 <ASP> Win3.1 req (2of2)

GOING COMMERCIAL

Bert Fischer

As some of you are aware through the ESC forum, we are bringing our first commercial program to the marketplace (doing all the work and distribution ourselves). One of the first thoughts you should have is: How do I want to package this product and how much space will it require (how many diskettes or CDs)? In our case the packaging will be a jewel case which will hold one HD diskette. This has provided us a very professional package.

Some drawbacks are

1. The manual can only be 24 pages (including cover) and the size is limited to approximately 4 1/2 x 4 3/4.
2. The jewel case will hold a maximum of two high density diskettes. If you need more room and a CD is not going to work for you than you must go to a box.
3. Most people are used to seeing a CD inside this type of container (it must be clearly labeled NO CD), although more companies are starting to use this method.
4. Room for additional information inside is limited.

Some advantages are:

1. The manual can have full color printing on the cover, which in turn is used for the cover of the container. Printed on a standard 8 1/2 x 11 page, they can be placed two up. This is less costly.
2. The back cover can be full color which can be printed two up on a standard 8 1/2 x 11 page.
3. The manual and diskette stay together in a nice small package for easy storage.
4. It's easy to ship in a small cardboard container.
5. Filler is not required to take up additional space when your manual and diskettes do not fill the box.

These are just a few things to think about when looking into the packaging. As more manufacturers start using a jewel case for floppy-based programs, they will become more accepted and familiar on store shelves (they are already starting to appear in stores like Egghead).

A few other things to think of are the individual costs of each item, color printing of manual covers, printing the manual, diskette labels and duplication, registration card printing, license agreement printing, color printing of a box, final assembly and shrink wrap. If you do not have the time to assemble the complete package, there are companies that specialize in supplying you with a complete package.

I'm sure by now you want some pricing information about using a jewel case. Depending on what you add to the package the price is anywhere from $2.50 to $5.50 per package in quantities of 2500. This would include a 24 page manual (full color cover), full color art on back, registration return card, license agreement, duplicated HD diskette, diskette label (two color), jewel case with floppy insert and shrink wrapped. It does not include the color separated film for printing, artwork or printing masters. These are separate costs which should be spread over the life of the product if nothing in the package changes.

Do not put things like version numbers on the printed package. If you must have a version number on your package have some small labels printed separately and stick them on the outside of the box before shrink wrapping. This will save on the cost of having new artwork

and reprinting new covers. Printed version numbers will not allow you to reuse any left over containers. Make your artwork on the package as generic as possible so it can be used over several versions or until you want it changed.

One other thing to look at is: What can be included in my package to make it a bit more attractive (if there is room)? In our case we have made arrangements with a font developer to include 10 Truetype fonts with our software. Now you are also faced with a royalty to pay to a third party for every package sold. This can become an accounting nightmare if you are not prepared. It does, however, add more value and appeal to the overall package.

Now that we have looked a bit into packages, printing, and additional files, let's take a quick look at advertising. This is where most of you will get sticker shock, if you have not done any advertising before. The basic cost per a 1/6 page B/W ad in a major magazine is approximately $800 per issue, depending on the length of time your ad stays in that publication. You can't say a lot of words in this space, so you have to try to say as much as you can and try to make it as positive as you can to attract customers. If you think you want to go with a full page, full color ad you are now looking in the neighborhood of $16,000 per issue. Ads between 1/6 page B/W to full page, full color fall somewhere between the above two figures. As you may now have noticed, advertising can get very expensive, so you must try to pick the magazines that you think will have the most benefit for your product. Also remember that when you place your ad in a magazine it normally takes two to three months before your ad appears (this may vary from magazine to magazine). Try to time the production of your product so it will be ready at least a month before the ad appears so you are ready to take orders and ship your product.

The initial cost to produce the first 2500 copies and have them delivered to your door step is going to be between 6 to 12 thousand dollars depending on the artwork, packaging and how much of the work you do yourself. This does not include advertising, and mailings. Be prepared to set aside about 200 to 500 copies of your first production run to send to distributors and reviewers. Don't forget, this is free advertising if you can get your program reviewed in a magazine.

Some things we have not covered. Do you have an 800 number to make it easy for customers to call using a credit card? Do you take credit cards? Who will take your orders? What should the retail price of your product be? What should the vendor price be? Where are the discount lines for quantities? How will you ship your product to the customer? Is your accounting set up for invoicing and returns? Do you have a place to store all those shrink wrapped packages you just paid for? Package artwork (don't scrimp in this area), press releases and reviews are also in there.

So as you can see, there are a lot of things to think about in bringing a product to the commercial side of the fence. However, you do have one thing working for you in that your shareware program can also be used to advertise your commercial program. If your shareware program is distributed widely, you already have some name recognition. You have an established customer base to start with. Send them a flyer of your new program and give them a discount. If they liked your shareware program enough to register they will probably purchase your retail version if given a good price. When establishing an upgrade price, consider that they have already paid you once for your shareware version. You should think about giving taking something close to the price they paid for the shareware registration off the upgrade price.

Beg, borrow or purchase mailing lists or labels. Most of all, be prepared to spend lots of time making contacts with distributors, stores and whoever or wherever else you think can sell your product.

- Main Menu

INTRODUCTION TO THE INTERNET

Rosemary West

The Internet (originally called ARPANET) began in 1969 as an experimental project for the Advanced Research Projects Agency (ARPA) of the U.S. Department of Defense. A network originally consisting of only four computers, its purpose was to allow communication between grant scientists. By 1972, 50 universities and military research sites had access to the network.

An important feature of the network's design was its ability to use many different routes among the computers, so that messages would not be limited to fixed paths to reach their destinations. If one computer broke down, the others could still talk to each other. So that all the computers could speak the same language, the designers developed a communications protocol called Transmission Control Protocol/Internet Protocol (TCP/IP), which became the Internet standard in 1983.

During the 1980s, many other public and private networks became interconnected through the Internet. Today, the Internet includes academic, government and commercial networks all over the world. With the increasing presence of local access providers and easy-to-use software, the system has become available to anyone with a personal computer. Current Internet features include email, FTP, Telnet, Usenet, Mailing Lists, Gopher, and World Wide Web.

- Email: One of the most important benefits of the Internet is email (electronic mail). From an Internet site or online service such as CompuServe, it is very easy to exchange messages with anyone else who is connected.

- FTP: File Transfer Protocol refers to the ability to transfer files from one computer to another, much like downloading from a BBS. Many FTP sites allow "anonymous" access, which simply means that you do not have to have an account with the host system in order to gain access to files.

- Telnet: Telnet increases the amount of access you have to a remote computer, allowing you to run programs on the host system.

- Usenet: Usenet is a collection of discussion groups, called newsgroups. People post articles and responses in the groups, and each day this material is copied to computers all over the Internet. You can select topics of interest and pick the articles you want to read.

- Mailing Lists: These are similar to newsgroups, but are sent through email to list subscribers.

- Gopher: This is a series of menus that lets you browse through Internet sites for access to information, files and utilities.

- World Wide Web: In a radical departure from the menu-based approach of Gopher, the Web is a series of hypertext pages which may include graphical images, sounds and animation. You click on highlighted words or pictures to jump directly to different information sources. The Web includes access to FTP and Gopher.

There are many software packages for browsing the Web. Many of them are shareware, or are included when you establish an account with an access provider. Popular titles include Mosaic, NetScape, and NetCruiser; these and others are often bundled with books about the Internet. If you have a CompuServe account, you can access the Net using CompuServe's NetLauncher software. America Online also has plans to introduce full Internet access to its subscribers.

Typically, access providers charge a monthly fee which includes a certain amount of online time, with additional time billed at an hourly rate. Sometimes the monthly fee includes a small amount of space for your own "home page". Many providers will also rent space for your own FTP site. Fee structures for commercial use vary widely and can run into hundreds or thousands of dollars per month, so if you are considering creating an Internet location for your business, shop carefully.

BACK IN BUSINESS

Andy Motes

Last year my shareware business was so bad that I decided to get out of the shareware business completely. My registration rate had lowered to a point that my new versions were bringing in about 15 registrations the first year. I decided there was nothing left to lose by dropping my ASP membership and crippling my software. I should have done it years ago.

I wrote a new version of School-Mom and distributed the complete program. However, I fixed it so that after 15 uses and 15 days it would quit. In other words, the customer has to have it for at least 15 days and have at least 15 uses before it quits. When it quits, a message appears stating that the initial evaluation period has ended and gives instructions for purchasing the software. After I receive the money I send the instructions for getting the program started again. The instructions include a registration number that is calculated from the owner's name. The owner has to type his/her name and this special registration number into the initial screen before the program will run again. After this is done once, a register.dat file is created and it's no longer necessary to enter the registration number -- the program will work forever.

I have had no complaints from customers. Actually, I've had lots of good comments about how fair and sensible it is. And more important, that new version that I distributed less than a year ago is now outperforming all my old versions that have been out for years. My registration rate is up and my overhead is lower because all I do is send a small instruction sheet with the customer's registration number. I no longer write shareware, I write trialware. I'm back in the software business.

TIP - DESIGNING WITH COLOR

The inventor of computer-based education software ends up wearing many hats. This person must become an instructional designer, market research analyst, programmer, psychologist, software configuration specialist, subject matter expert (sometimes affectionately called a "SMEE"), animator, and graphics designer.

The key to graphics design is selectivity. The color design of a product should integrate with the objective of the product. Some colors invoke symbolic or emotional responses. Color harmonics is a mathematical formulation of two-color combinations to define the best combinations.

Do not try to use all fonts and colors just because they are available. Sometimes just black and white make sense. Don't produce an Angry Fruit Salad unless this is your intent. AFS is an interface design that uses too many colors, suggesting a bizarre canned fruit cocktail.

Lauren Gascoigne

Vol 1, No 2, 1992

- Main Menu

THE NEXT CENTURY

If your software uses dates, it is not too soon to start re-designing it for the year 2000.

Most current PC's will accept system dates between 1980 and 2099, and many applications have the ability to handle a much broader range of dates. However, the software's ability to "understand" that a year is outside the 1900's usually depends on its being entered as a four-digit year, rather than as a two-digit abbreviation.

Try this at your DOS prompt: Use the DATE command to enter 12-31-2095. The next time you enter the DATE command, your PC will probably tell you it's Saturday 12-31-2095. But if you enter the DATE as 05-11-95, DOS will assume you mean Sunday 12-13-1995. If you enter 01-01-2050, DOS will accept it, but if you enter 01-01-50, it will tell you that's an invalid date, because it assumes you mean 1950, and it won't take a date earlier than 1980.

Except for software that deals with historical dates, your programs probably expect users to enter the date in two-digit rather than four-digit format. Even if the system date is correctly set to 2000, when you enter 00 as the date, most of these programs will either reject it or assume you mean 1900.

Another problem will be date sorting and range checking. If your program sees 12/31/99 and 01/01/00, it will assume that the "00" date comes before the "99" date. This may not be true.

This is going to cause a lot of problems for a lot of people unless they get their software updated between now and then. There's plenty of time, but for some businesses using large, complex systems, this may be very expensive. It may also become an issue for shareware authors whose old versions continue to circulate for many years. For those who have not done so already, now is a good time to start releasing versions that can cope with the new millennium.

Rosemary West

TIP - SHARING INCENTIVES

This is one area where we can really help each other. A nice incentive is to include in your registration packet a copy of another educational shareware program written by an ESC member. Another is to provide a coupon for $5 or $10 off the registration price of another program by an ESC member.

Exchange shareware disks with another member and include it in your registration packet along with the coupon. Remember, always give a time limit for how long the deal is good. If the customer has an incentive to act immediately he is more likely to purchase.

Andy Motes

SPACE-SAVING TIP

I was working on a program that saves "setup" and "high score" files to disk and it struck me that I had a lot of little files eating up disk sectors. I converted them to one large setup file with over 50 entries and a master "high score" file. The result was a single disk sector (2k on my hard disk) for each file. This is much less than the 30 or so files of 2k each. The code was simpler also. Remember that DOS lists the size of the file when you ask for a DIR but not the actual disk space allocated.

John Gallant

Vol 1, No 6, January 1993

- Main Menu

CD-ROM AUTORUN

Under Windows 95, it will be possible to have a CD-ROM start playing automatically as soon as a user pops it into the drive. Everyone should add this capability to both DOS and Windows CDs now, by including a text file called AUTORUN.INF in the root directory of each CD.

The first line of the file is the word "autorun" in square brackets, like this: [autorun]

The second line is the name of the program or batch file to be run, followed by optional command parameters. For example:

```
[autorun]
open = mygame.exe
```

This will cause mygame to run when the disk is placed in the CD-ROM drive. You can add additional parameters with the "shell" command to create a Windows menu. For my CD, I've used these lines:

```
shell\go = &Go
shell\go\command = menu.bat
shell\nosound = &No Sound
shell\nosound\command = nosound.bat
shell\help = &Help
shell\help\command = notepad readme.txt
```

This creates a Windows menu that looks like this:

Go
No sound
Help

If the user chooses Go, then my menu.bat runs. If the user chooses No Sound, then my alternate nosound.bat program runs. If the user chooses Help, then notepad is invoked with my README file. The & in front of the letter defines a shortcut key (e.g. user can type G, N or H).

Note that "shell\go = &Go" defines the menu item and shortcut, while "shell\go\command = menu.bat" attaches a program to the menu. The words after shell\ (go, nosound, help) can be anything you want as long as you use them both in defining the menu item and attaching the program to the menu item. SHELL and COMMAND are the reserved words. Don't use the word "menu" immediately after "shell\".

It is good practice to have your application come up and immediately offer the user a way out. Ideally, you should have a title screen come up with three choices at the bottom: Setup, Play, Quit. That way users can get out fast if they don't want the program to start.
Karen Crowther

Vol 4, No 3, June 1995

- ESC CD-ROM
- Main Menu