ED 330 315 IR 014 934

AUTHOR	May, Charles
TITLE	The American Short Story: From Poe to O. Henry. A HyperCard Application.
INSTITUTION	California State Univ., Long Beach. English Dept.
PUB DATE	90
NOTE	38p.; Supported by the 1989-90 Dissemination Grant from the California State University Lottery Revenue Program for Instructional Development and Technology.
PUB TYPE	Guides - Non-Classroom Use (055)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	Authoring Aids (Programing); *Computer Assisted Instruction; Computer Software Development; Elementary Secondary Education; English Instruction; English Literature; Higher Education; *Hypermedia; Short Stories
IDENTIFIERS	Apple Macintosh

ABSTRACT

This report describes a computer-assisted instructional application created on a Macintosh computer using HyperCard software. The instructional program is aimed at those who teach college-level English education courses and those who are planning a course on the use of technology in the English classroom. It is noted that the HyperCard software was developed to aid in teaching English literature, specifically short stories, and provides access not only to the text of the short story, but also to concepts and patterns throughout the story. The rationale behind using the Macintosh computer, the concept of hypermedia and hypertext and, in particular, the use of HyperCard on the Macintosh, are discussed. Also described is the theory of short story analysis that underlies the computer application. The report concludes with a detailed discussion of programming using the HyperCard software, and suggests a method for creating unique applications to meet the needs of individual classrooms. (DB)

The American Short Story: From Poe to O. Henry

A HyperCard Application

By

Charles May

English Department
California State University, Long Beach

The American Short Story: From Poe to O. Henry

A HyperCard Application

by

Charles May
English Department
California State University, Long Beach

Distributed Free to CSU English Faculty

1989-90 Dissemination Grant
Funded by CSU Lottery Revenue Program for
Instructional Development and Technology

Introduction

American Short Story: From Poe to O. Henry is a computer-assisted instructional application created on a Macintosh computer using HyperCard. In its most basic format, it was developed as an instructional device to teach the rudiments of HyperCard to teachers and future teachers in English 337--*Technology in the English Classroom*, a CSULB course created to fulfill Title 5 requirements that students receiving the clear teaching credential have computer training specific to their discipline. The preparation of the present version of the application and this booklet was supported by a 1989/90 dissemination grant from the Lottery Revenue Program for Instructional Development and Technology. The application is being distributed free to all interested English faculty in the California State University system; it is particularly aimed at those who teach English education courses and those who are planning a Title 5 computer-instruction course.

I am not a computer programmer, nor a designer of computer applications, as this "rough draft" version of *American Short Story* will indicate. However, by studying instructional books on HyperCard, I have developed a basic format whereby HyperCard might be used to assist the teacher of English teach the reading of short narrative. HyperCard, and its program language, HyperTalk, are very accessible devices for designing simple applications such as this one. The booklet and disk are being distributed to my colleagues in English Departments throughout the CSU system to let them know that useful applications can be designed even by a computer novice like myself, for with HyperCard, one does not have to learn a complex program language. Although it is impossible to provide a complete background for the use of HyperCard in this short booklet, perhaps an account of the thinking and development which went into the creation of this simple application will provide even those who are unfamiliar with the Macintosh or with HyperCard with an idea of the program's possible educational uses. With that aim in mind, I will briefly discuss the following elements which underlie the *American Short Story* application:

1. The "look and feel" of the Macintosh computer
2. The basic concept of Hypertext
3. The implementation of Hypercard on the Macintosh
4. The theory of short story analysis which underlies the application
5. How the *American Short Story* application works
6. How the *American Short Story* application was put together

The actual HyperCard program cannot be shipped with this disk; thus it is assumed that you have access to a copy of it. Having been distributed free with all Macintosh computers since August 1987, it is ubiquitous. If you do not have your own copy, find a Macintosh lab on your campus, and it will surely be there, probably on a fixed disk or on a network file server.

Although this application makes use of ten nineteenth-century American short stories, it is not being distributed primarily for use by teachers of the American short story, but rather for use by teachers of English education courses. The emphasis is not on the history of the form, but on analysis of individual stories. Short stories were chosen because I have done a great deal of research and writing on the form over the past several years; these particular American short stories were chosen because they are relatively short, are fairy well-known and representative, and are in the public domain. Teachers of the short story are, of course, free to use the application in their classroom. You may make as many copies of the disk as you like; however, I would appreciate it if you would give credit to me for designing the application and to the Lottery Revenue Program for Instructional Development and Technology for supporting it. I am currently preparing a short story text book, which, when published, will be distributed with disks similar to this one.

The Macintosh Computer

The Macintosh is manufactured by Apple Computer, Inc. which was founded by two young entrepreneurs, Steve Wozniak and Steve Jobs, in the early 1970s. The Apple I computer, created in a garage, was followed by the Apple II, which became the computer of choice for public school systems throughout the country. Apple, wanting to develop a more powerful machine to appeal to the business market, created a computer in the late 1970s called the Lisa. However, because it was so expensive and because IBM had practically cornered the business market with its computer by that time, the Lisa failed to take off. Steve Jobs then pioneered the transformation of the Lisa into what he envisioned as the people's computer--the Macintosh--a computer "for the rest of us" as the ad slogan went, which anyone could set up and use "right out of the box."

The Macintosh was introduced in 1984 to mixed reviews. The public took to it immediately, for it seemed the ultimate in "friendliness," right down to the opening "happy face." However, the old hands at computer use--who had cut their teeth on the CPM world of cryptic commands lines, considered it an expensive toy, not fit for serious work. That has all changed now, and the machine has been upgraded and expanded through several different models since then. The Mac SE is only slightly faster and can be purchased with an internal fixed disk drive with a 20 megabyte capacity. The SE30 runs about four times faster than a plain SE and, in addition to being provided with a fixed disk drive of 20 megabytes or more, has a 1.44 Megabyte internal "superdrive" which can also access data created on an IBM-compatible computer. The Mac II series constitutes the so-called high end of the Mac family. There are several models, all of which support color monitors and all of which are faster (except for the now discontinued "plain" Mac II), more "open" and more easily expandable than the relatively closed Mac Plus, Mac SE, and SE30.

Two primary factors distinguished the Macintosh from other computers when it was first introduced: the use of the mouse and a graphic user interface (now widely imitated and called a GUI) Before the Mac, the basic way that one "made things happen" on a computer was by typing in instructions on a typewriter style keyboard. This seemed quite natural, given the fact that the most common uses of the computer involved manipulating words and figures--by word processing, data base management, and spreadsheet programs. However, the creators of the Macintosh felt that such a "command line" orientation frightened many potential computer users away because they had to memorize a large number of keystrokes. Steve Jobs wanted to develop a "user interface" (what one sees on the screen and uses to operate the machine) which would be more "friendly" because it featured images of familiar "real world" objects. Thus, the first thing you see when you turn on the Mac is not the enigmatic A> prompt of the IBM, but a "desktop."

Although the "desktop" is simply a gray opening screen, the graphic images on it, appropriately known as "icons," resemble real objects. For example, if you start up the Mac with a 3 1/2 "floppy" disk, you will see an icon or graphic image of that disk (with its name under it) at the top right corner of the screen. You will also see the image of a trash can at the bottom right of the screen, where you "dump" things you no longer want to use on the desktop. The logic of the disk icon is that it "contains" things you wish to use; thus you must "open" it and spread them out. However, before you can open the disk icon, you must first "select" it.

To select an icon, you must make use of that other major Macintosh innovation--the mouse--so called because of its size and because it has a "tail" of sorts (the wire that connects it to the computer). The mouse is an analog device; as you roll it back and forth across a table top, a marker (usually a pcinter) moves across the screen desktop to correspond to the mouse movement on the actual desktop. There is also a button on the top of the mouse, conveniently located so that if you place your palm over the mouse body, your index finger can be poised over the button. You "select" the disk icon by "pointing" to it (that is, moving the mouse until the arrow or pointer is resting on the icon) and by "clicking" on the mouse button once. You "open" the icon by immediately clicking on it a second time.

When you "double click" the disk icon, you will see spread out on the desktop a group of additional icons, usually of three kinds: program icons, file icons, and folder icons. Program icons and file icons vary depending on what kind of program they represent; e.g., a word processing icon may be a quill pen, and a file created with a word processing program may look like a sheet of paper. Again, it seems quite reasonable (what interface designers like to call "intuitive") that you would open a file folder to get at what is in it. You do so by "double clicking" the mouse button to simultaneously "select" and "open" the folder. You may see other folders inside a folder which can be opened in turn. You will also see icons for programs or for files created by programs.

To start a program you need to click on it twice rapidly in succession (double-click) or else make use of the menubar at the top of the screen. The menubar, where functions of what Apple calls the "Finder" are activated, lists the following words: **File, Edit, View, Special.** If you put the arrow on one of these words and hold the button of the mouse down, a "menu" will drop down. For example, if you point to the word

File and hold the button down, you will see such words as **New Folder, Open, Close,** etc. on the menu. Still holding the mouse button down, if you "drag" the mouse across the table, bringing the pointer or arrow down the menu, a dark bar moves down the menu also, "selecting" the functions one by one. If you release the mouse button while the word **Open** is dark, you will start the program you previously selected. (If you select and open a file, e.g. a "page" icon representing a file you have created with your word processor, you will simultaneously open the word processor and load the file selected into it.)

The Concept of HyperText

HyperCard is the best known and most popular implementation of a category of computer software known generically as Hypertext. Although HyperCard has only been in circulation for a couple of years, the concept of Hypertext has been around since the mid 1940s when Vannevar Bush, President Franklin Roosevelt's Science Advisor, described a hypertext system called "Memex" which used microfilm technology. Recognizing that the human mind works by association rather than by straight linear progression, Bush emphasized his proposed system's ability to link seemingly disparate items together. However, the real father of Hypertext, the man who coined the term in the 1960s, is Ted Nelson, who exploited the computer's capacity for presenting natural language in a non-linear way to create a hypertext system he called "Xanadu," named after that "magic place of literary memory" in Coleridge's dream poem "Kubla Khan." Nelson's effort is indeed literary, for his aim is to put books on-line in hypertext format so that links can be established between them in numerous ways.

Before the advent of Apple's HyperCard, the best known hypertext system was "Notecards," developed at Xerox PARC. One of the principle designers of the system was Randall Trigg, who wrote a Ph.D. dissertation at the University of Maryland in 1983 entitled *A Network-based Approach to Text Handling for the Online Scientific Community*. Calling his system "Textnet," Trigg envisioned a system which would permit collaboration between critics commenting on a primary text, with various links identifying different types of comments, e.g. comments that support, refute, or indicate the relevancy or inadequacy of the points made in the text. "Notecards," which grew out of the "Textnet" idea, is primarily a Xerox in-house application developed to help analysts gather information and produce reports. Based on the metaphor of the 3x5 notecard, the program allows the user to display a number of notecard-like windows on the screen and to develop links between them of various types.

Other well-known hypertext applications are Brown University's "Intermedia" system, which has been used to teach an English literature course focusing on traditional literary history; "Project Jefferson" at University of Southern California, which has been developed to support the Freshman writing program; and the "Perseus" project at Harvard University, which is an attempt to collect and organize vast amounts of data about classical Greek civilization with elaborate links between text, images, and sound.

HyperCard on the Macintosh

HyperCard, a hypertext program created for the Macintosh , is difficult to explain unless one has the program up and running. However, some general comments may be helpful to get an overview of its features. Although HyperCard is not a word processor, it does allow you to type text within it. Although it is not exactly a data base manager, it does allow you to retrieve information easily and quickly. Although it is not a spreadsheet, it does organize information in interrelated blocks. HyperCard is more than any one of these three common computer applications; it is a highly flexible environment, driven by an easy-to-learn programming language, for connecting and relating information and creating applications. Hypercard was created primarily by Bill Atkinson and has been supplied free with all Macintosh computers since late 1987. For those who purchased Macintosh computers before that date, it is available from Apple Computer, Inc. for under $50.00. Atkinson has said that he insisted to Apple that the program be "given away." And although it is a very powerful program, making it available free is just good business; Apple is banking on the hope that people will buy a Macintosh because it is the only machine which will run HyperCard.

The HyperCard Program Icon is the image of a stack of cards with a hand holding a pencil poised over it. The files created by HyperCard are in the image of an identical stack of cards without the hand poised over it. These "Stacks" (the key word for the organizational object of HyperCard) are files which can be stored in folders just as any other Macintosh file. Since HyperCard's introduction in 1987, enough two-inch-thick books have been written about the program to fill a ten-foot long shelf. However, the manual released with the program by Apple, on the other hand, is only a relatively thin wire-bound document. The reason for this "minimal manual" is that Atkinson and Apple did not want to scare users away by providing such daunting documentation of all the program's features. The manual which comes with HyperCard does indeed provide enough information for one to "use" the program, but those who wish to exploit its power to create their own applications can indeed discover more information by going to one of the more detailed books listed in the bibliography.

Although working through the later discussion in this booklet on how the *American Short Story* application was created may be the best introduction to HyperCard, a few words here about how it is structured may be helpful. If you have never tried the program and wish to explore it on your own first, start it by double clicking on the HyperCard icon (a hand poised over a stack of cards), and you will be taken to the Home card where you will see a group of icons for several "sample" stacks which Apple has included with the program. Click once on the icon labeled **Intro.** (Note: although you double click on icons "outside" of HyperCard to "open" them, you click only once while inside HyperCard.)

HyperCard is an information storage and retrieval program. It is much more than that, but that is as good a place to begin as any. The difference between HyperCard and other information storage and retrieval programs, such as database management programs, is that HyperCard will let you store information in such a way that you do not have to access it linearly, one thing after another in a predetermined order; rather you can access information by association because of the way HyperCard "links" items of information together.

The simplest example would be an electronic "book" on HyperCard stacks. When you see a footnote number, you could simply click on a button at that point and the actual footnote reference would pop up on your screen; you would not have to turn to the end of the chapter or look down at the bottom of the page to find it. I know that does not sound like much, but the ability to access additional information immediately --simply because it has some connection with the information at hand-- offers several interesting possibilities, not the least of which is to create a truly intertextual framework for whatever primary text you are dealing with.

[illegible] place where you store information in HyperCard (the equivalent to a file in usual computer terms) is called **a stack.** the metaphor is a stack of notecards, in which each individual screen full of information in HyperCard represents a single **card.** However, the stack is not a linear stack; it is more like a ring of keys or a rotary file. When you thumb through the stack and get to the last card, the next card starts the first one again.

HyperCard is made up of five basic components:

Stacks
Backgrounds
Cards
Fields
Buttons

The stack cards have at least two levels: a **background** level and a **card** or foreground level. Let's say you have a stack of 3x5 index cards which you use for bibliography entries. Each card has the following slots for information on it:

Author:
Title:
Place of Publication:
Publisher:
Date of Publication:

This format on each card is to remind you what goes in each slot. Because this five-item form or template is on each card, we could call it the **background level**; because the information you place in the various slots will change from card to card., we could call the level of individual authors and titles you would enter the **card level.**

Fields are those areas on the card where you can enter text. They are blank slots waiting to be filled. For example, on HyperCard you might want to create a separate field for each of the five items above: author, title, publisher, etc. By doing so, you could then use HyperCard like a database manager and search for information on any of the fields.

Buttons, however, are the most interesting of the HyperCard elements. Whereas a **field**, a **card**, and a **stack** have real-world equivalents in 3x5 index cards, there is no equivalent for a **button**. Think of it like this: Let's say you are looking at a bibliography card which has a reference on it to another book similar in some way to this one, maybe with the familiar command *SEE*.... However, instead of shuffling manually through your stack of cards to find that cross-referenced item, what if you could point your finger at it and the reference would suddenly, almost magically, appear. This is what a **button** in HyperCard can do. It is the primary means by which you create cross references and links between different cards in a stack or between different stacks.

For example, in the *American Short Story* stacks, if you click on the button that has the little light bulb on it, a field will suddenly op up with some information on it. If you click on the button that says **Hide**, the button sends a message to the field to go away. If you click on the button with the hand poised over a sheet of paper writing, you will be taken to a card somewhere else in the stack which presents a writing prompt for the student to consider and write about. If that writing prompt mentions the word **irony** and it is outlined in a box, you can click on that word (for it has a button hidden behind it) and be taken to the Terms stack for a card which provides a definition of irony.

HyperCard has five different levels which range from limited access to the program's features to full access. These can be set by going to the **User Preference Card.** To get there, start from the **Home stack** and use the left arrow key to cycle back through the **Home** stack until you find the **Preference Card.** Set your own copy of HyperCard to the highest level by clicking on the little button by the word Scripting. Even though you may not use it right away, this level is the only one that gives you access to the HyperCard scripts. If you want to look at the scripts which activate the buttons in the **American Short Story** stack, you must be at this level.

HyperCard is powered by a program language called **HyperTalk**. If you once tried to learn Basic or Pascal and gave it up in despair because it seemed like a foreign language, rejoice! **HyperTalk** is close enough to English that you not only can read the scripts behind the buttons and understand most of them readily, you can often make a guess about what kind of commands HyperTalk will understand. Moreover, the very fact that **HyperTalk is a modular or object-oriented** language means that all the instructions you create are small groups of lines (called **scripts**) "attached" to separate **objects** (most often **buttons**). Thus, you do not have a long list of numbered lines to keep track of. You can work with small sections of instructions at a time.

If you are fascinated by what computers can do, you will love working with HyperCard, especially when you feel confident enough to experiment with HyperTalk. But even if you are not so enthralled with computers, HyperCard has the ability to change your mind, for it will show you how to manipulate the knowledge you deal with every day in ways that may dramatically assist both your acquisition of that knowledge and your communication of it to your students.

Theory of Short Narrative Analysis

Desktop, and now laptop, machines have demystified the computer in the past five years by taking it out of the mainframe basement and the hands of the computer wizards and making it available for the "rest of us ." Computers are everywhere on university campuses now: IBM-type machines cranking out figures in the business department and Macs designing graphics in the art department. However, in spite of the fact that by far the largest number of desktop computers are used for the manipulation of ordinary language, English department faculty--specialists in the use of language both ordinary and extraordinary--have been the slowest to make extensive use of the machine. Of course, most faculty engaged in research have a computer with which they do their writing, maybe even using it to log-on to large remote data-bases to develop their working bibliographies. And a number of English departments have developed computer writing labs where students are supervised to do their

500-word essays using word processing software. However, that's about as far as it goes. And indeed, one might ask, "what else is there?" A number of faculty members who very early on made use of computer-assisted instructional drill programs were quickly disenchanted with what was simply electronic versions of grammar drill handbooks, and so they went back to their own holistic and open-ended methods of teaching reading and writing.

However, it is possible that with the advent of hypertext, particularly the versatile and easy-to-use HyperCard, the computer can help us teach the reading of literature--that is, once we are agreed on what it means to teach the reading of literature. The most extensive empirical research and theoretical studies on literature in the past decade have focused on the reading of narrative. For literature teachers, and even more particularly for their students, the basic paradox that plagues the teaching of narrative is that it is a linear form which must ultimately be read in a non-linear way. The problem is more pronounced in the teaching of the short story than it is in the teaching of the novel, for the length of that loose and baggy monster makes reading it any other way than linearly difficult. Whenever we ask students to read stories, they dutifully do so (sometimes) in a straightforward linear way--scanning the letters, words, sentences, paragraphs, translating them into action, watching things happen, hearing people talk, trying desperately to get to get to the end of the story And the "end" for them means merely the conclusion of a series of events which occurred "one damned thing after another."

However, when we ask them what happened in the story and they begin telling us the events one damned thing after another, we say, "No, that's not what I meant, that's not what I meant at all." Then we show them that something "happened" in the story which they did not see. How can that be? Given a fairly attentive student reader who can summarize the story events accurately, how can it be that he or she did not see the same things happen which the teacher did? The problem arises, of course, with our assumption that the narrative has meaning--that it is, its narrative "end" or conclusion makes possible an epistemological, psychological, metaphysical, or other discursive "end" or purpose. The narrative, we somehow tenaciously believe, somehow stands in lieu of some other kind of statement, even if we are not always quite sure how it does this.

We know the story has some rhetorical structure with which it communicates its rhetorical purpose, but it does not quite seem the same as other rhetorical patterns. Students are taught early on that other kinds of writing use rhetorical modes which make use of certain conventional devices to replicate different means of arriving at knowledge. Definition , for example, arrives at knowledge by placing a new term within a category and then differentiating it from other members of that category; Classification arrives at knowledge by sorting a group of objects or phenomena in such a way as to elicit generalizations about what characteristics they share. Yet, we are not sure how the rhetorical mode of narrative arrives at knowledge-- except perhaps by providing us with an illustration or an example. However, since the notion of illustration reduces the story down to the mere exemplum, one of the most primitive narrative forms, we still do not understand how more complex narratives arrive at knowledge or how we as readers elicit knowledge from them.

The problem centers on what we mean by "to read" when we ask a student to read a story for tomorrow's class, for the student seld understands that there are at least two meanings of the verb "read. First, there is the "first" reading, when one, not having read the story before, discovers what happens in it; second, there are "subsequent" readings, when one processes the story, already knowing what happens next. The "first" reading is the kind that reader response theorists seem most interested in, for one can "read" a story the first time only if he or she has the appropriate previous knowledge or schemata to bring to it. E. D. Hirsch is, of course, well known for insisting that people can read only if they have "cultural literacy," which for him, means knowing a whole list of hard data. (As will be shown, HyperCard does provide a relatively painless way to provide this kind of knowledge network for students.)

A great deal of research is going on right now in the psychological community on the nature of discourse processing: how do we read? what is a story? However, on digging into this vast amount of material, we find that matters are still on a rudimentary level with the cognitive scientists--they are still trying to determine how people process a story in such a way that they can recall its events or trying to find out just what the minimal units of a story are--neither of which are of any real help to us in trying to teach students how to read Joyce's "Araby."

Finally, of course, there are the structuralist and post-structuralist narratologists, with their attempts to make the reading process a scientific endeavor by abstracting stories and reducing them down to their most basic aspects of grammatical categories. The results of this approach have been such detailed studies of such minute elements of narrative that they were exhausting to read, and the few teachers who did read them were reluctant to expose their students to them.

It is only now with deconstruction that we have returned to interpretation and analysis of the individual work. Although the painfully sophisticated analysis or interpretation of a story the way Jacques Derrida or J. Hillis Miller practices it may exceed what we want to do in the classroom, at least deconstruction has returned our attention to an analysis of the work as it is laid out before us all at once, rather than as we process it one damned thing after another in time. Furthermore, by forcing us into looking at the very process by which fiction comes into being as fiction, deconstruction forces us into a self-conscious, self-reflexive mode in which we must concentrate on "how" the work "works," not on what "happens" in it or what it "means."

Reading is a bottom-up/top-down process. First we attend to the details, attempting to formulate an overall conception of the total events of the story piece by piece; however, once we have the whole thing in mind (at least in terms of its surface elements of events, characters, speech, etc,) then we proceed top-down--moving from this macrocosmic design back down through the individual elements of the work to try to determine if there is a hidden nonlinear pattern lying there somewhere covered over by the surface linear action.

Looking at narrative in terms of its pattern rather than in terms of its events is as old as Aristotle, of course. But it got its most influential boost when it was adopted by the Russian and American formalists of the 1920s and then by such latter-day formalists with a mythic difference as Joseph Frank and Northrop Frye in the 1950s. Later, such structuralists as Claude Levi-Strauss, building on the early Russian Formalists, tried to develop a method for "reading" story (which Levi-Strauss sees as synonymous with myth) by reorganizing the mere events into stacks or bundles of themes or motifs related to each other on the basis of function or similarity. What Joseph Frank and Northrop Frye meant by spatializing the temporal nature of the story, Levi-Strauss extended by transforming the temporal context of the events spread out through time into a spatial code existing all at once. Moreover, when Roland Barthes broke up Balzac's novella *Sarrasine* into 561 separate but related uni'ꞩ of meaning he called "lexias," similarly, he was transforming the story from a temporal contextual pattern of events into a unified entity traversed by various codes.

The text becomes an intertext, for it can only be read in terms of the various coded conventions, both literary and cultural, which make it meaningful. Umberto Eco has called this process by which a text must be read intertextually a "net-labyrinth." In a "net" every semiotic entity in the work (every word, sign, symbol, motif) is connected or linked to numerous other semiotic entities in the story (and outside it) by associative paths. To "read" a story from this point of view means to move from node to node by pathways of similarity, opposition, contiguity, etc. Eco says to read a net is to explore the pathways of an unlimited territory.

In this respect, literary theorists are interested in the same thing that information theorists are intert ed in--how reading is the processing of rhetorical codes which have developed from actual reading practice and which have become conventionalized. Moreover, both argue that although readers internalize these codes by practice, in order for their reading skills to reach higher levels of understanding, and in order for those skills to become increasingly automatic, they must become conscious of these coded conventions; they must achieve a "metaliterary" awareness of what they are doing when they read. A primary way this can be achieved is, as they Russian formalists argued a half century ago, for the "devices" of the work to be "laid bare" or "foregrounded." In this way, the student can be exposed to what gives the literary work its "literariness."

Although I cannot develop these theoretical ideas about the reading of narrative in any more detail here, it seems clear from this brief summary that hypertext with its concept of linked associative paths is an ideal means by which to "objectify" the way we read narrative intertextually. What I have tried to do in the *American Short Story* stacks is to make use of HyperCard to create connections or links in a small group of stories so that they might be read intertextually as nets. The aim is to expose the underlying "literary" or conventional unity of the work. My modest application only scratches the surface, but it may be sufficient to give others ideas to develop further.

The American Short Story: What it Does

The American Short Story application contains ten stories arranged on HyperCard fields. The student accesses these stories from a Table of Contents Page shown below. By simply pointing to the title of one of the stories and clicking on it, the student will be taken immediately to the first page of the story.

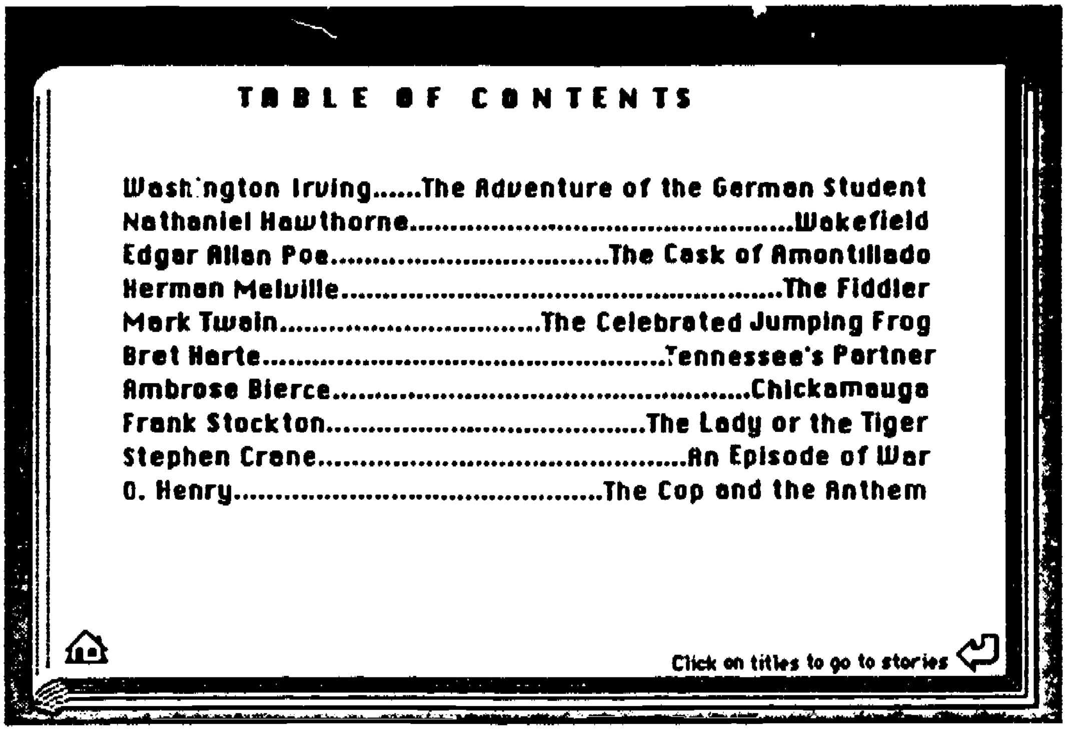
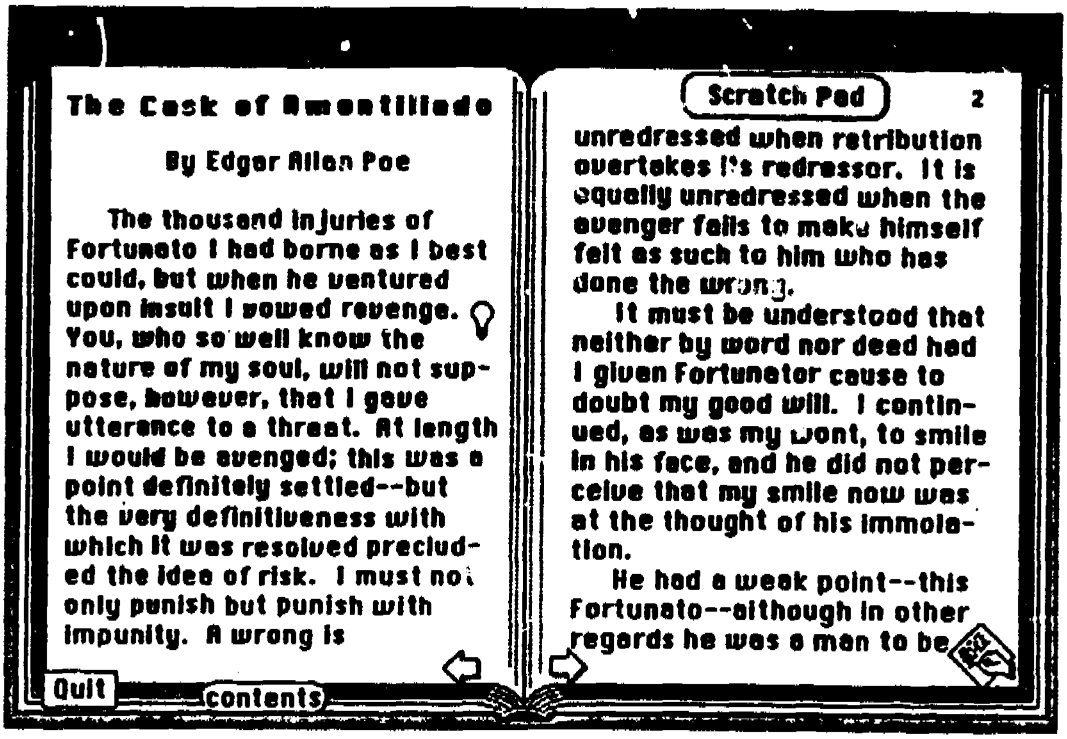


TABLE OF CONTENTS

Washington Irving	The Adventure of the German Student
Nathaniel Hawthorne	Wakefield
Edgar Allen Poe	The Cask of Amontillado
Herman Melville	The Fiddler
Mark Twain	The Celebrated Jumping Frog
Bret Harte	Tennessee's Partner
Ambrose Bierce	Chickamauga
Frank Stockton	The Lady or the Tiger
Stephen Crane	An Episode of War
O. Henry	The Cop and the Anthem

Each page of the story is a separate background field, with two pages on each HyperCard screen or card. The following is the first card of Poe's "Cask of Amontillado." It contains four buttons: the light bulb activates a pop up window of information. The hand writing on a sheet of paper sends the reader to a Writing Prompt Card shown . (See the following two figures). The Writing Prompt has a button that will return the reader back to the page of the text where he left it.



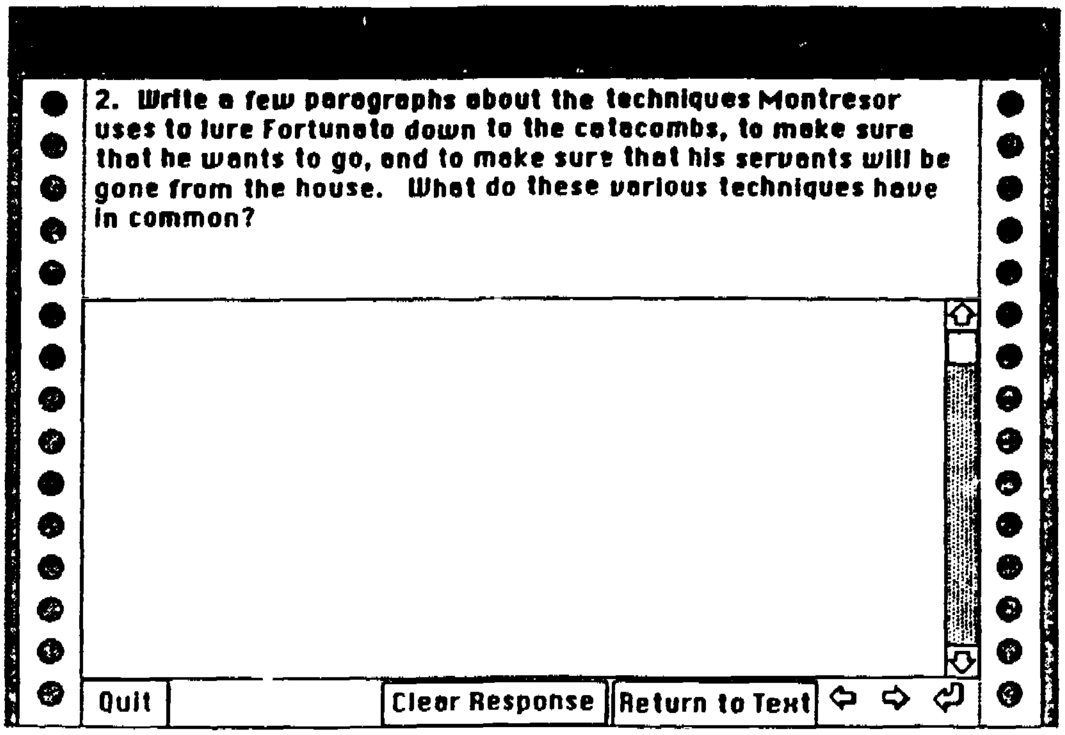
The Cask of Amontillado

By Edgar Allan Poe

The thousand injuries of Fortunato I had borne as I best could, but when he ventured upon insult I vowed revenge. You, who so well know the nature of my soul, will not suppose, however, that I gave utterance to a threat. At length I would be avenged; this was a point definitely settled--but the very definitiveness with which it was resolved precluded the idea of risk. I must not only punish but punish with impunity. A wrong is unredressed when retribution overtakes its redressor. It is equally unredressed when the avenger fails to make himself felt as such to him who has done the wrong.

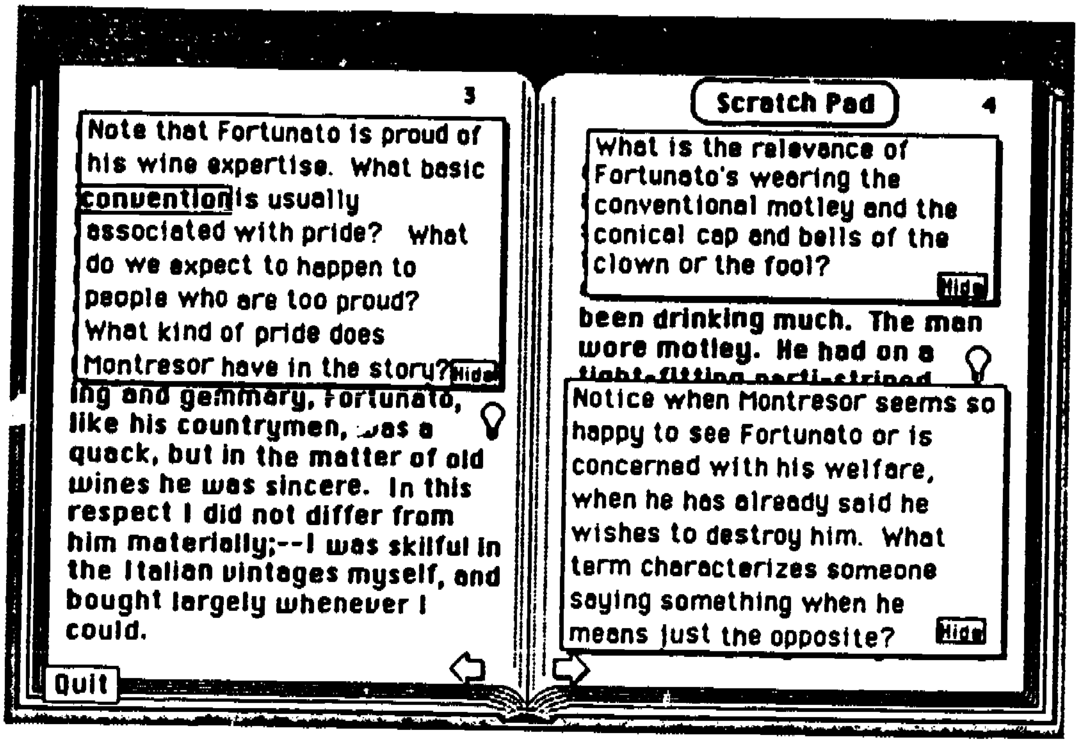
It must be understood that neither by word nor deed had I given Fortunator cause to doubt my good will. I continued, as was my wont, to smile in his face, and he did not perceive that my smile now was at the thought of his immolation.

He had a weak point--this Fortunato--although in other regards he was a man to be



2. Write a few paragraphs about the techniques Montresor uses to lure Fortunato down to the catacombs, to make sure that he wants to go, and to make sure that his servants will be gone from the house. What do these various techniques have in common?

The following is the second card of the "Amontillado stack"; it shows three pop up windows. The little squares with **hide** on them are also buttons. If the student clicks on one of them, both the pop up field and the **hide** button will disappear until the next time the light bulb is clicked on. Note that the first pop up window has the word **convention** in boldface with a box around it. This is also a button. If the student clicks on it, he or she will be taken to the card in the **Terms Stack** with a definition of **convention.** Notice that this card also has the word **Formalist** in a boldface box; if the student clicks on it, he or she will be taken to the **Formalist** card in the **Terms Stack.** Pressing the crooked arrow at the bottom right corner will return the student back to where he or she started.



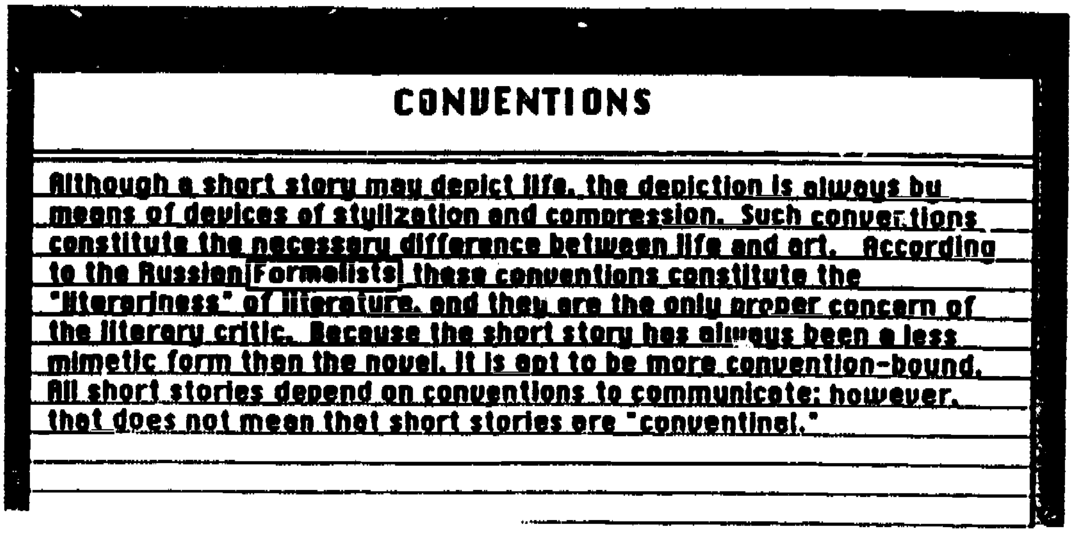
Note that Fortunato is proud of his wine expertise. What basic convention is usually associated with pride? What do we expect to happen to people who are too proud? What kind of pride does Montresor have in the story?

ing and gemmary, Fortunato, like his countrymen, was a quack, but in the matter of old wines he was sincere. In this respect I did not differ from him materially;--I was skilful in the Italian vintages myself, and bought largely whenever I could.

What is the relevance of Fortunato's wearing the conventional motley and the conical cap and bells of the clown or the fool?

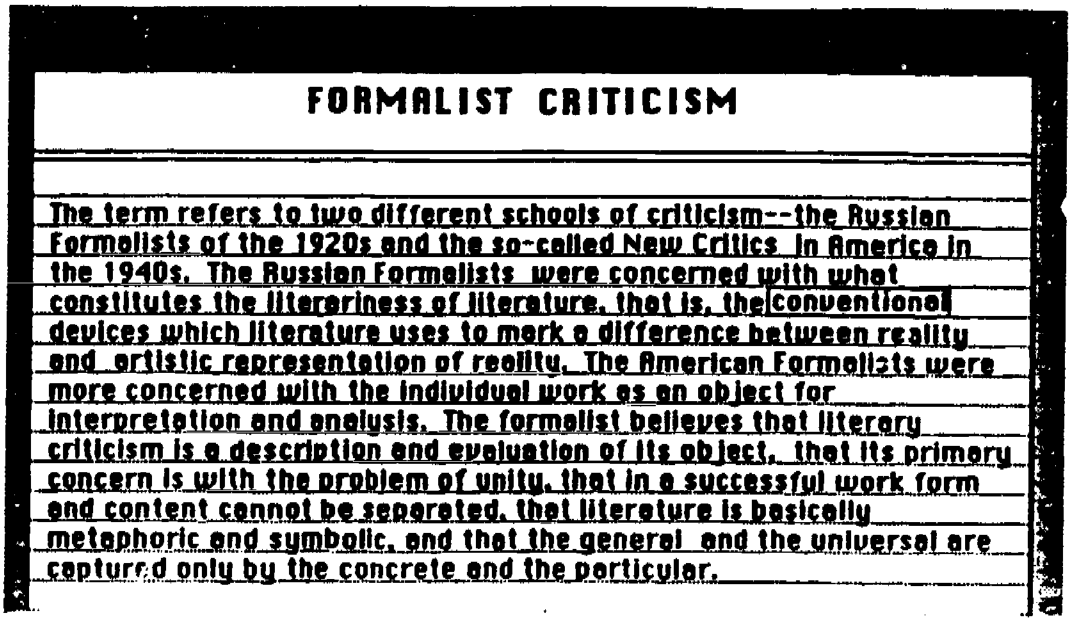
been drinking much. The man wore motley. He had on a tight-fitting parti-striped

Notice when Montresor seems so happy to see Fortunato or is concerned with his welfare, when he has already said he wishes to destroy him. What term characterizes someone saying something when he means just the opposite?



CONVENTIONS

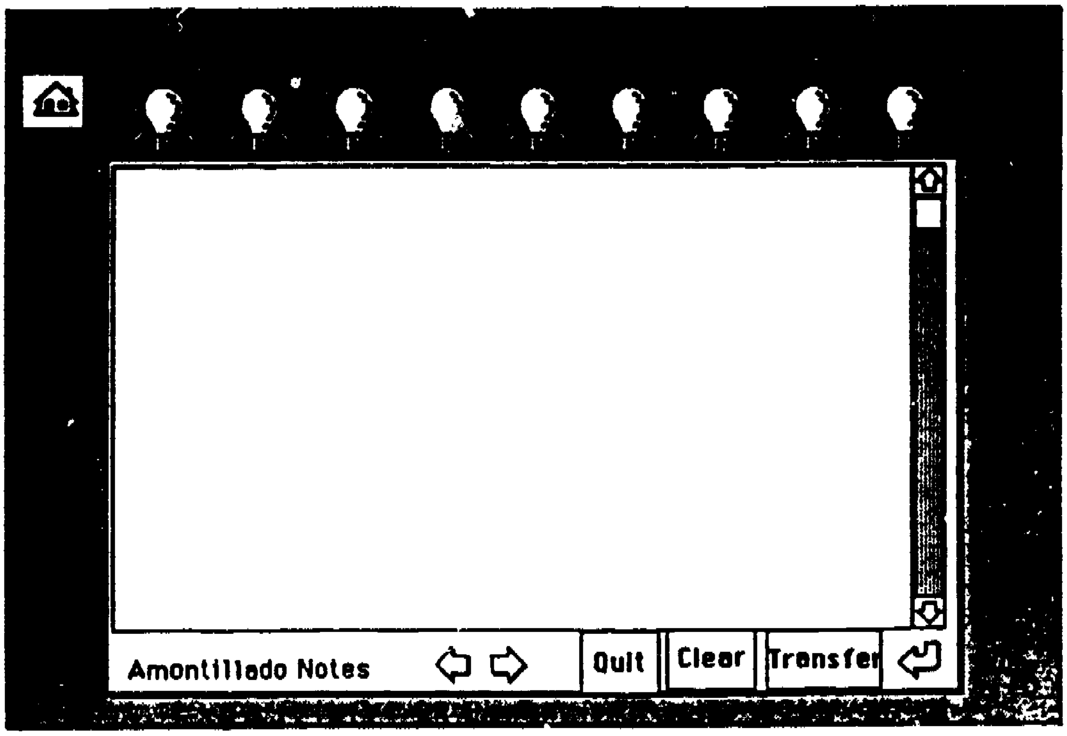
Although a short story may depict life, the depiction is always by means of devices of stylization and compression. Such conventions constitute the necessary difference between life and art. According to the Russian Formalists these conventions constitute the "literariness" of literature, and they are the only proper concern of the literary critic. Because the short story has always been a less mimetic form than the novel, it is apt to be more convention-bound. All short stories depend on conventions to communicate; however, that does not mean that short stories are "conventinal."



FORMALIST CRITICISM

The term refers to two different schools of criticism--the Russian Formalists of the 1920s and the so-called New Critics in America in the 1940s. The Russian Formalists were concerned with what constitutes the literariness of literature, that is, the conventional devices which literature uses to mark a difference between reality and artistic representation of reality. The American Formalists were more concerned with the individual work as an object for interpretation and analysis. The formalist believes that literary criticism is a description and evaluation of its object, that its primary concern is with the problem of unity, that in a successful work form and content cannot be separated, that literature is basically metaphoric and symbolic, and that the general and the universal are captured only by the concrete and the particular.

The Scratch Pad button will take the student to the following scratch pad where he or she may jot down notes on what the pop up windows have made him think about. The student returns to the text again by clicking on the crooked Return arrow at the bottom right corner. The Quit button on all the cards exits the HyperCard program. The contents button takes the student back to the table of contents.



Both the Scratch Pad and the final Writing Prompt card in each Story stack have a **Transfer** button on them. When the student clicks on this button on the Scratch Pad, all the notes in the Scratch Pad will be placed in a folder called **Notes**, which the student can then access with a Macintosh Word processor such as Microsoft Word, Word Perfect, Microsoft Works, etc. Moreover, by clicking on the Transfer button on the last Writing Prompt Card, all the writing prompts will be gathered up from the separate Writing Prompt Cards and put together as separate paragraphs or blocks of text in a folder called **Essays** which the student can access to reorder, rewrite, and edit on his or her word processor. The student can cut and paste items from the Amontillado file in the Notes folder and put them in the Amontillado file in the **Essay** folder. In this way, the student's responses to the pop up windows and the Writing Prompts can serve as the brainstorming basis and the outline framework for a paper on the story.

Setting up Your Own Story Stacks

The following suggestions for setting up the short story stack assumes you have the full HyperCard program, with its accompanying **Home** and **Idea stacks.** Although there are public domain, shareware, and commercial stacks available which might make your own stacks easier to create and more attractive, I have stayed completely with the materials provided with HyperCard. What follows is a step-by-step description of the creation of the *American Short Story* stacks. You can modify your own stack as you see fit. I have used "amontillado" as an example.

Creating The Story Stacks:

Getting the Background Graphic and Entering Text

From the **Home Card**, click once on **Stack Ideas** and search through by clicking on the pointing finger at the bottom right until you find the **Open Book 3** card design. Click on it once to go to that design. It will have information already on it.

From the **File** menu choose **New Stack.** When you are asked to name it, give it the name of your story, e.g. Amontillado; click on the word New. You now have a blank **Open Book 3** Card which serves as the background of the text of your Amontillado stack.

Pull down the **Tools** menu and click on the **Field** tool (it is the top right icon and looks like a small box with lines in it) . You will note there are already two fields on the Open Book card. Simply resize them to fill most of the two "pages" on the card by clicking on one of the sides or corners of the field and "dragging" the mouse away from the field. (The field is "selected" when it has a "marquee" effect around the sides--what is known as "marching ants."

Double click on one of the two fields to get to the **Field Info** dialogue box. Click on font and choose Chicago as your font for the field, 14 as your size, and 16 as your line height. If the size choice you want is not available, delete the existing number below the fonts size choices box and type in your choice. Click O.K.

Do the same thing for the second field. Now everything you type in the two fields will be in this basic font. Experiment with other font styles, sizes, and line heights if you want. I have chosen Chicago for its screen readability.

You can begin typing in your text now by clicking an insertion point near the top right corner of your field. **Remember**, when you get to the end of the first "page" or field, you must set an insertion point at the top of the next field before you can begin typing there. Moreover, you should proofread the text in each field before going on. Changes affect only subsequent text in each individual field. For example, word wrap only works within each field. Thus, if you find that you have left out a line on "page" one after you have finished typing in page two, you may not have room left in the field of page one to put it in. When you get to the end of the second field, press **Command-N** (the Command key is the one just to the left of the spacebar) and you will have a new blank card of double-sided pages on which to type.

The *American Short Story* application provides three means by which the student "stops" reading in order either to think about certain issues or to write about certain issues. The first is a simple pop-up window activated by clicking on a light bulb button. The window makes suggestions the student might think about. If he or she wishes to jot down some ideas on the suggestions, he or she can click on the Scratch Pad button to go to a scratch pad for writing down ideas. The second stopping point, indicated by the image of a hand writing, is a more explicit writing prompt whereby the student must write a paragraph in response to a question about the story. You must create a separate card for the Scratch Pad and the Writing Prompt. Then you must create buttons to go back and forth between the text and these two new cards. Here is how you create the cards and the buttons.

Getting the Graphic For the Scratch Pad

Go back to the **Home** card by pulling down the **Go** menu and choosing **Home.** From the **Home stack** go to **stack ideas** and find a graphic called **Ideas;** click on it once to go to it. It will have text on it.

Pull down **File** menu and choose **New Card.** You will now have a new blank card with a row of light bulbs and the words IDEAS AND INSPIRATIONS across the top.

Pull down the **File** menu and choose **Cut Card.** The new card you just created will be saved in the Macintosh Clipboard. Pull down the **Go** menu and choose **Recent;** you will see a kind of catalogue (reduced in size) of all the cards you have been to or looked at recently. Find the last page of the story stack you created and click once on it. You will be taken there.

Pull down the **File** menu and click on **Paste Card;** your new card will be pasted in a slot right after the last page of your story.

Pull down the Objects menu and click on **card info.** When you see the card dialogue box, type in a name for this card; call it **Scratch Pad.**

Pull down the **Edit** menu and click on **Background;** this is where the graphics of the card are. (Note the slash marks around the menubar; this lets you know you are in the background.) Pull down the menu, choose the eraser (the rectangle object in the third row center of the tools) and erase the words IDEAS AND INSPIRATIONS by holding down the mouse button while dragging the eraser over the words; leave the row of light bulbs.

Deselect the **Background** by pulling down the **Edit** menu and clicking on it again. Pull down the **Tools** menu and select the **Field** mode. Select the field by clicking on it; stretch it to cover the space below the row of light bulbs. Double click the field to get to the **Field Dialogue box.** Choose **Scrolling** as the Style. Click on **Font** and choose Courier 12, line height 14. (The general rule is to make the line height two points higher than the font size; however, you can experiment with different combinations.)

Creating the Scratch Pad Button

Now you need a button that will allow the student to go to the **Scratch Pad** card anytime he or she has an idea to jot down. You only need to create this button once. Since it should appear on every card or screen of your story, you must put in the **Background** level.

Go to the first page of your story; pull down the **Edit** menu and choose **Background**; a jagged edge or slash marks will appear around the Menubar. Pull down **Objects** menu and choose **New Button.** Drag the new button to the top of the right hand page (or wherever else there is room and); then double click on it to go to **Button Dialogue Box.** Name it **Scratch Pad;** make sure that Show Name is on (has an x in the button by it) and that round rec is chosen as the style of the button.

Click on the word Script and insert this line between the **on mouseUp** and **end mouseUp** lines:

go to card "Scratch Pad".

(Don't forget the quotation marks) Click O.K. There is no real need to create a Return to Text button; the scratch pad stack already has one in the bottom right corner (Remember, however, this will only return the reader to the card where he was immediately before he went to the Scratch Pad.) Pull down the go menu and go **Home.**

Creating the Writing Prompt Cards

From the **Home** card, choose **Stack Ideas** and find a graphics titled **Computer Paper;** click on it once to go to it. Pull down the **Edit** menu and click on **New Card.** Pull down the **Objects** menu and go to **Card Info.** Name the card **Writing Prompts.** Now pull down the **Edit** Menu and choose **Cut Card.** This will cut the new blank card from the **Ideas Stack** and put it in the Clipboard. You now have a blank piece of computer paper. You need to create two fields on the computer paper: one at the top where you can type in the writing prompt itself, and a scrolling field on the rest of the page where the student can type in his or her response to the prompt. Here's how to do it:

Pull down the **Tools** menu and choose the **Field** tool and you will see that there are two fields already there; you just need to resize the top one to make it large enough to hold the prompt.

Name the top field **Prompt**; make the font Chicago, 12, with 14 line height. Name the bottom one **Response**, make it scrolling, and make the font Courier 12, line height 14. This was the same font and line height you made for the **Scratch Pad.** Later, when you want to merge the information you have typed in the Scratch Pad with information you have typed in the **Prompt Response**, the fonts will be the same. Click **O. K.**

Making the Pop-Up Fields and Buttons

For each Pop-up field that appears to provide the student with additional information, you must create one new Field and two new Buttons. Pull down the **Objects** menu and click on **New Field.** Drag the field to wherever you want it on the screen; double click it to get the field dialogue box and make it opaque; click on **show lines**; it will be superimposed over top your text field.

Pull down the **Objects** menu and click on **New Button.** Make it transparent. Do this once more to make another button. One of these buttons will be the button the student will see to make the pop-up appear; the other will be used to hide the pop-up field when the student has finished reading it.

Pull down the **Tools** menu and choose the **Button** tool to see where your two buttons are . Drag one of the buttons to some place in the margin where you where you want to identify a passage the student is to think about. Drag the other button inside the new field you have created. Double click this second button to get to the Button dialogue box; name the button **Hide.**"

Go to the script of the **hide** button and type in the following between the **On mouseUp** and **End mouseUp** lines:

Hide card field id (whatever the field number is)(Press Return)
Hide button id (whatever the button number is)(Click O.K.)

(You can find out what the field id number is by choosing the **Field** tool from the **Tools** menu and then double clicking inside the field to get to the field dialogue box. You can find out what the button id number is by choosing the **Button** tool from the **Tools** menu and double clicking the button to get to the button dialogue box.

For the second button I have used a small graphic of a light bulb to mark when I want the student to click. You can go the stack **Art Ideas** and find it on a card called **Miscellaneous 1**.

Pull down the **Tools** window and use the lasso to choose it by pulling the lasso around the light bulb to make a complete circle. When it has marching ants, go the **Edit** menu and choose **Copy Picture**. It will be saved in the Clipboard.

Go back to your story where you want the first idea prompt to appear and pull down **Edit** and choose **Paste**. The light bulb will appear. While it is selected (marching ants), drag it to where you want it and then drag the transparent button over top of it and size it to fit. Go to the button's script and type in the following lines between the **on buttonUp** and **end buttonUp** lines (use the same field id numbers and button idea numbers you used above):

Show cardfield id (whatever the field number is)(Press Return)
Show button id (whatever the button number is.(Click O.K.)

You now have your first Pop-up brainstorm idea button, with a pop-up field and a button to hide the field when the student has read it.

At this point, while your light bulb (or whatever you decide to use) is still in the clipboard, it is a good idea to go through your whole story and choose **Paste** from the **Edit** menu wherever you want a pop-up window to appear. When you are finished pasting in all the icons you can simply go back and create your pop-up buttons and fields, transposing one of the transparent buttons over each light bulb. Remember the light bulb is a graphic separate from the button itself; thus, if you decided to move one, you must move the other one separately to go with it. This is different from using buttons with icons already on it; in this case both the button and the icon will move together, as shown below with the Writing Prompt buttons.

Making the Writing Prompt Buttons

The writing prompt buttons have an icon superimposed over them in the form of a hand writing on a sheet with a pencil; this is the traditional ico ı for a word processing program; we will use it to indicate that the student has to write something here. To create the first one, go the page where you want the student to stop and write and pull down the **Objects** menu and select **New Button.** Click on it twice to get to the Button dialogue window. Click **off** the Show Name and leave the button at round rect.

Click on the **Icon** button and a window of icons will appear. Double click on the icon with the hand writing and you will be taken back to your story where the icon will now be in the button. Resize the button and position it where it will not interfere with the text.

Double click to get to the Button Dialogue window, click on **LinkTo.** With the LinkTo window on the screen, go to the end of your story where your first writing prompt card is located. When you get there, click on **This Card.** Your Writing Prompt button will be linked to this card and you will be taken back to the card where your Writing Prompt button is located.

When there, pull down **Objects** menu and select card info and jot down the card id number. Use your new Writing Prompt button to go to the Writing Prompt page.

Pull down **Objects** menu and choose **New Button.** Name this **Return to Text** and make it a round rect with the name showing. Click on Script and between the **on mouseUp** and **end mouseUp** lines, type

go to card id (type the id no. of the card where your prompt button is).

Now you can move back and forth from your Writing Prompt button in the text to the Writing Prompt Card where students are asked to write. Write your prompt in the top field; your students can use the scrolling field for their response. After you make the first Writing Prompt button, you can copy it (by going to the **Button** tool and then choosing **Copy Button** from the **Edit** menu to put it in the clipboard; then going to the page where you want it and choosing Paste Button from the Edit Window.) You can also copy the **Return to Text** Button on your other Writing Prompt cards

DON'T FORGET, HOWEVER, TO CHANGE THE SCRIPT OF EACH BUTTON TO INDICATE THE APPROPRIATE CARD TO WHICH IT IS TO GO. YOU CAN DO THIS EITHER BY USING LINKTO OR BY TYPING THE DESTINATION CARD ID NUMBER IN THE BUTTON'S SCRIPT)

Creating the Table of Contents Stack:

Getting the Graphic and Creating a New Stack

From the Hypercard **Home** card, click once on the **Stack Ideas** Icon. Click your way through using the pointing finger icon until you find an icon entitled simply **Book Page**. Click on it once and you will be taken to a card design that has a U.S. map and some information about the state of Nebraska on it. This is the background graphic used for the Table of Contents card. It was used instead of the "Open Book" graphic used for the text of the stories because all the titles will fit on one card easily.

To make your own Table of Contents stack (which will consist of only one card), pull down the **File** menu and select **New Stack.** You will be taken to the dialogue box used to save files, and the cursor will be flashing at the point where you may enter your stack name.

Type in the word **Contents.** Make sure that the file directory just above indicates that you will be saving the new stack on the drive and in the folder where you want it. Use the Drive button or the subdirectory button for your hard disk to open the folder where you want to save **Contents.**

Click on the New button. What happens is that a copy of the "Book Page" graphic is placed in your new **Contents** stack with all the information about Nebraska erased from it. Now you can set up your contents page the way you want it. (One of the nice things about HyperCard is the ease with which you can make use of preexisting graphic designs; Apple has provided several with the program for your use.)

Creating a New Field

The first thing you need to do is create a field to hold the list of authors and titles you will type in. To do this, pull down the **Objects** menu and click on **New Field**. A small rectangular field will appear in the middle of the screen. The outer edges of the field will be shimmering in what is known as the "marquee" effect or "marching ants." This means that the field is "selected." Use the mouse to point on any one of the edges of the field and drag the mouse away from the field to enlarge it. To move the whole field around, click anywhere inside it and drag. To reduce the size of the field, click on a corner and drag in toward the center. Once you have the field large enough to cover the page, click outside the field to freeze it into place

Choosing Fonts for your Field

The font size and style of the text you will type into the field must be set for the field as a whole. To do this, pull down the **Tools** menu and select the **Field** tool; click inside the field once to select it (create the marching ants).

Pull down the **Objects** menu and select **Field** Info. (You can also achieve this by double clicking in the field.) The cursor will be flashing in the **Field Name** blank. Type in the word **contents.**

Select the style as **Transparent;** click the show lines off (that is, if the box has an x inside it, click on it to remove the x and deselect that feature). To set the font, click the font button once and you will be presented with a font dialogue box. You can choose what you want by clicking on one of the font names and setting the size you want in the size box. You can also choose a style such as boldface, italic, etc. and you can align the text in the field in one of three different ways. The *American Short Story* stack, for example, uses Chicago 14 (If the size you want is not listed in the scrolling field of font sizes, simply delete the size indicated in the box below it and type in what you want. The box with the word "sample" shows you what you will get). When you get everything set as you want it, click on OK and you are returned back to your field. (At present, you cannot mix font styles within a single field. Thus, you cannot underline a word in a field without underlining everything, that is, unless you create a separate field for it. It is hoped that Apple will fix this in the next release of HyperCard.

Entering Text

Now to enter in the Table of Contents information. Pull down the Tools menu and click on the **Browse** icon. (You cannot type in a field while you are in the Field mode, only in the Browse mode.)

Create a Table of Contents title by using the paint tool. Here's how: Pull down the **Tools** menu until it detaches from the top and place it anywhere in the field. Double click on the large letter A to get the font dialogue box again. You want a larger size than 14 for the title, but it is probably best to stay with the same font style. The *American Short Story* stack uses Chicago 20, bold, center align for the heading TABLE OF CONTENTS . Select what you want and click O.K. Type in TABLE OF CONTENTS at the top of your field where the cursor is flashing. Close the Tools menu window by clicking on the tiny Close box at the top left corner.

Move the cursor below the Table of Contents Title near the left margin. When the small hand becomes a vertical bar, you are in the field. Place the bar as close to the top left of the field as you can and click once to start an insertion point. Now you can type in your titles. At the end of each line, press Return and type another until all are done.

Creating Buttons

You can now create buttons for the Table of Contents which will take the reader to the first page of each story and return him or her back to the contents page if necessary.

To do this pull down the **Objects** menu and select **New Button**. A new button with the name New Button will appear in the middle of the screen. Double click on the button to go to the button dialogue box. Choose transparent and click off the box Show Name. Click O. K. and you will be returned to the Contents Page.

Now stretch the button by clicking on its corners the way you did the field to make it long enough and narrow enough to exactly cover up the first title on your contents page. Once you have done this one time, you can "clone" this button by holding down the option key and dragging the button down; a duplicate button will appear just below. If you hold the shift key down also while you do this, the button will move down in a straight line. Clone a transparent button over all your titles. You may need to stretch some of them to make them cover the whole title.

Once you have finished this, you will create a simple one-line script to activate the button to make it go to the first page of your story. Here's how: Hold down the Option and the Command key (the Command key is the one with the apple on it just to the right of the Option key) simultaneously and click once on your first title button. You will be taken to a script of that button. It will have nothing on it but two lines **on mouseUp** and **end mouseuUp**. Type the following between the two lines:

Go to card one of stack Amontillado (or whatever the name of the stack is that you want to go to).(click O.K.)

Do this for each one of your titles. You need to use the exact title of the stack (which may or may not be the full name of the story; you must be exact here; for example, if you type *Go to card one of Stack Cop and Anthem*, you will get an error message if the stack is entitled *Cop & Anthem*

Now you need to create a button which will return the reader from the text to the Table of Contents. Use the Table of Contents button you have just created to go to the first page of your first story. Pull down the **Edit** menu and choose **Background**. Note the slash marks on the menubar which indicate you are in the Background.

Pull down the **Objects** menu and choose **New Button**. Double click on the button to go to the Button Dialogue Box. Title this button **Contents**; turn on the Show Name and make the button round rect. Click on **script** and type in the following command between the **on mouseUp** and **end mouseUp** lines:

go to card one of stack Contents.

Pull down the **Tools** menu, choose the **Browse** tool and try the **Contents** button out. It should take you back to the Table of Contents page. The button you put over the story title Amontillado on the contents page should take you back to page one of the story "The Cask of Amontillado."

Now all you have to do is copy the **Contents** button to the background of the first page of all of your stories. To do this, go the first page of your first story where the Contents button is.

Pull down the **Tools** menu and choose the **Button** mode; click on the **Contents** button to choose it. Pull down the **Edit** menu and choose **Copy Button.** Go to the first page of your second story.

Pull down the **Edit** menu and choose **Background.** Then choose **Paste Button** . Your Contents button will be pasted on the background and thus will appear on every card in your second story stack. Do this for each one of your stories.

Tidying Up

You now have a Table of Contents with buttons on each story title ·· hich will take you to that story's first page; you also have a button on every screen of each story which will take you back to the Table of contents page. Your Table of Contents page, as it is in a book, is the central point of reference for your story stacks. To finish tidying up, delete the two arrow keys at the bottom of the Table of Contents card; since this stack only has one card in it, pressing those buttons will do nothing and might confuse your reader. You might want to keep the Return key at the bottom right; it simply takes you to the card where you were immediately before you went to the Table of Contents card. You can also delete the Home card at the bottom right if you want; I have left it so that you might more easily go back there if you want. I have also left the menu bar at the top of the card for navigation purposes. However, you can remove it if you don't want your students to use it by pulling down the Objects menu, clicking on Stack Info and then Script and adding the line *hide menubar* in between the on openStack and end openStack lines. If anytime you want it back just hold down the Command key and press M to get the message window and type show menubar.

Creating The Terms Stack: Choosing the Graphic

From the **Home** card, go to **Stack Ideas** and browse through until you find **index card 3**, the largest index card graphic in the Stack Ideas; it has information about a bicycle sprocket chain on it.

Pull down **File Menu** and choose **New Stack**. At the save dialogue box, name this stack **Terms**; make sure it is on the disk or in the subdirectory on your hard disk where you want it saved, and then click on the **New Stack** button. The bike information will be removed and you will have a blank index cards with lines on it. If you pull down the **Tools** menu and click on the **Field** tool you will see that the card actually has two fields on it; that's o.k. We will use the small field at the top for the name of the term and the large field at the bottom for the definition of the term.

Copying the Cards

However, before we start entering information, let's duplicate a number of these cards while this one is still blank; it will save time later. Simply pull down the **Edit** menu and choose **Copy Card**. Pull down the **Edit** menu again and you will see the line **Paste Card**; beside it is a symbol of the command key and the letter V; this means you can hold down the command key and press the letter V simultaneously to activate the Paste Card command. Do this for a number of times; each time you press the letter V while holding down the Command key, you will create a new copy of the blank index card. You can check how many you ave by pulling down the **Objects** window and choosing **Stack Info.** The Stack Info dialogue box will tell you how many cards the stack contains. (Remember, the HyperCard stacks are not stacked linearly, but in a chain or ring like a ring of keys or a rotary card file. The card after the last card is the first one).

Choosing Fonts

Choose the **Field** tool from the **Tools** menu and double click on the small top field; when you get to the Field Info dialogue box, click on Font and choose a font for the name of the Term; it probably should be larger than the font for the actual definition. Let's stick with Chicago throughout to avoid confusion. Type in 16 for the font size and choose bold to make it stand out; choose center for align. Make the font for the definition text field Chicago 12, with line height 14, left align.Click O.K.

Go back to the card and choose the **browse** tool from the **Tools** menu. Type in the name of the Term at the top, e.g, *Allegory*; press Tab to go to the large field, (or else use the mouse to create an insertion point at the beginning of the field) and begin typing in your definition.

Transfer Buttons

The final buttons on the *American Short Story* stacks are **Transfer** buttons on the **Scratch Pad** and on the last **Writing Prompt** card within each story stack. These buttons gather up the student's notes from the **Scratch Pad** or his paragraph responses to the prompts on the **Writing Prompts** and transfer them to files within folders on the disk. Scratch Pad notes for "Cask of Amontillado," for example, are transferred to a filed named "Amontillado" in a folder named Notes. Writing Prompt responses are transferred to a file named "Amontillado" in a folder named Essays. You can copy these buttons to your own stacks if you wish. Or you can look at the scripts by choosing the button mode from the Tools menu and then double clicking the button, and choosing script to look at them and adapt them for your own use.

Now your story stack has the following elements:

1. Text cards with the full text of your stories entered on them in fields.

2. pop up buttons within the story to make the reader stop at certain places to think about elements of the story, especially the themes that are manifested as a result of repetition.

3. Writing Prompt buttons within the story which direct the student at certain points in the story to write a paragraph or two.

4. A scratch pad which the student can get to at any time to jot down notes on his or her reading of the story.

5. A transfer button that will let the student gather up all his or her writing prompts in once place and transfer them into a file for his word processor.

6. Another transfer button that will let the student transfer all his or her notes from the Scratch Pad to a word processor.

7. A stack of literary terms which the student is directed to wherever one of the terms is mentioned in the story.

Obviously HyperCard has much more potential than I have used in this simple application. For example, you can search for words or strings of words within each story stack by press Command-F (for Find) and typing in what you want to find; it is a good way to determine the repetition of key words or phrases in a story. However, it is impossible to do more than hint at the possible uses of HyperCard. Moreover, my own minimal instructions here may confuse you more than help you. However, once you grasp the overall concept and structure of HyperCard (which only comes from experimenting with your own stack ideas), you will be creating your own applications to meet the needs of your classes.

I would very much appreciate hearing from you about this stack or these instructions. I have tested them out, but somehow errors always manage to creep in. Please write me or call me and let me know how the stacks are working:

Charles May
English Department
California State University, Long beach
Long Beach, CA 90840
(213) 985-4218

Books On HyperCard

These are the books I consulted while preparing this application. There are many more waiting at your local library or bookstore.

Goodman, Danny. *The Complete HyperCard Handbook.* N. Y.: Bantam Books, 1987. This is the best introduction to HyperCard.

----------. *HyperCard Developer's Guide.* N. Y.: Bantam Books, 1988. Primarily covers how Goodman created the commercial stacks Focal Point and *Business Class.*

Goodman, Paul. *Power User's HyperTalk Handbook.* Blue Ridge Summit, PA: Windcrest Books. A manageable and helpful introduction to writing scripts for HyperCard.

Scafer, Dan. *HyperTalk Programming.* Indianapolis, IN: Hayden Books, 1988.

Schell, Barry. *Running HyperCard with HyperTalk.* Portland, OR: MIS Press, 1988.

Swaine, Michael. *Dr. Dobb's Essential Hyper Talk Handbook.* Redwood City, CA: M&T Books, 1988.

The Waite Group. *The Waite Group's Tricks of the HyperCard Masters.* Indianapolis, IN: Hayden Books, 1989. Includes scripts for various kinds of stacks by a number of HyperCard scriptors.

Weiskamp, Keith. *Mastering HyperTalk.* N. Y.: John Wiley & Sons, 1988.

This booklet was created on a Mac SE using Word Perfect word processing software. It was printed on an Apple LaserWriter Plus, using Times 12 and photocopied.