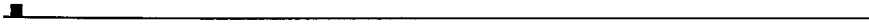


Graphic Design Archive Prototype 3.0



Graphic Design Archive

Lester Beall Collection

Rochester Institute of Technology

A thesis submitted to the faculty
of the college of Fine and Applied Arts
in candidacy for the degree of

Master of Fine Arts

by **David J. LaMarca**

May 9, 1990

Approvals

Advisor: **James Ver Hague**

Date: 7.1.90

Associate Advisor: **R. Roger Remington**

Date: June 1, 1990

Associate Advisor: **Chris Comte**

Date: June 1, 1990

Special Asst. to the Dean for Graduate Affairs: **Philip Bornarth**

Date: 9/14/90

Dean, College of Fine and Applied Arts: **Dr. Robert Johnston**

Date: 9/14/90

I, David J. LaMarca

hereby grant permission to the Wallace Memorial Library of the

Rochester Institute of Technology to reproduce my thesis in whole or

in part. Any reproduction will not be for commercial use or profit.

Date: 6/7/90

Dedication

This book is dedicated to my parents,
for sacrificing so much to get me where I am today.

I Love You!

Contents

Introduction	<i>VI</i>
History of the Graphic design Archive	<i>1</i>
Interactive Media	3
Laser Video Technology and it's role in the Interactive world	5
Prototype 2.0	8
Life of Lester Beall and the Beall Collection	10
Nodes and Networks	12
Prototype 3.0	14
Conclusion	21
Sources consulted	22
Appendix A: Posters	
Appendix B: Scripting	
Appendix C: Flow Charts	
Appendix D: Supporting Papers/Correspondences	
Appendix E: Synectics Session on the GDA	
Appendix F: Prototype 2.0 Screen Designs	
Appendix G: Prototype 3.0 Screen Designs	
Appendix H: Apple Computer, Inc. <u>Human Interface Guidelines</u>	
Appendix I: IRIS Intermedia	
Appendix J: "A Guide for Interactive Interface Design"	

Introduction

We live in an information age, as the book MegaTrends points out. This is an age where computers comprise most of the world's markets, defense, and even education. Historically, computers have been able to handle large amounts of data usually in the form of text or numbers. Modern computers have the ability to do this with even greater efficiency, but today text and numbers are not all that they are able to hold. Imagery is the newest addition to the information age. With the advent of laser video disc technology, a computer can be used to access and account for hundreds of thousands of images.

Video disc technology has improved greatly in the past recent years, today a single 12" disc can hold up to 108,000 still video images. A disc can also hold motion pictures in addition to still frames. These images can be accessed in any order at amazing speeds. New strides are being made everyday to make laser discs perform even better, and hold much more video.

The Graphic Design Archive is an ongoing project at the Rochester Institute of Technology which involves many different departments, faculty members, and student volunteers. The current laser disc holds over 25,000 graphic design images. The laser video disc was completed by the American Video Institute (AVI) at RIT, taking nearly two years to complete.

The computer software used to drive the laser disc is an enhanced Hypercard® stack. Hypercard® is a computer program developed by Apple Computer, Inc. and used on an Apple Macintosh® computer. It is used by developers as a vehicle in which to write software. In this case, Prototypes 1.0, 2.0, and 3.0 were all

Introduction

created in this way. In my thesis, Prototype 3.0, all of the interface designs were created by myself, along with most of the programming. As decisions were made concerning the visual interface, careful attention was made to follow Apple Computer, Inc.'s human interface guidelines of simplicity and consistency, (see Appendix H).

The workstation required to run Prototype 3.0 consists of a Macintosh computer with at least 2 mb of RAM, a color monitor and at least an 8-bit videocard and a hard disk drive. On the video end, a laser disc player capable of interfacing with a computer is required, along with a video monitor for viewing the images on the disc■

History of the Graphic Design Archive

The Graphic Design Archive (GDA), directed by R. Roger Remington, has been in existence at R.I.T. for five years now. More pieces are coming to the archive every year, mostly by donations. The purpose of putting the archive on a computer based system was simply to help organize the material in a way that would aid in retrieval of not only the items in the archive, but also the data about the items. The current laser disc of the archive holds approximately 25,000 images of pieces in the archive. While many of these images are in slide form, the majority of the imagery is taken from the actual archive original which is housed at RIT.

Sorting through this many images is indeed a chore made much easier with the use of the computer. Over the past three to four years the computerized version of the archive has taken on some major changes. The original program was written for the IVIS computer system, far from the much more eloquent Macintosh system which is in use today. This early version was pioneered by several different faculty members with various expertise. Four members of this original group are still very active in the GDA today: Roger Remington -Graphic Design, Mark Collien -American Video Institute, Steve Kurtz -Computer Science, and Chris Comte -Computer Science. Also there have been others involved such as Barbara Polowy from the Wallace Memorial Library and Susan Preston-Mauks, a graduate student from the College of Liberal Arts.

There have been many students who played major roles in the development of the Graphic Design Archive software packages (Prototype 1.0 - 3.0). For the past two years the GDA has been undertaken as thesis projects, and although

the main purpose may have been similar, the thesis certainly were not. Each graduate student who undertook the project as a thesis has brought his/her own originality to it. There have been many interface designs, as well as tools implemented into the software. The interface design of Prototype 3.0 is based upon Apple Computer Inc. Human Interface Guidelines. These guidelines state that the interface should resemble something which the user is accustomed to dealing with. For instance, in an office setting a person might use files, folders, and trash cans, therefore so should the computer interface which he/she uses. I have tried to incorporate this type of thinking into the design of Prototype 3 by making the tools, and design as intuitive as possible.

Apple Computer, Inc.'s human interface guidelines also state that a software program should be consistent with the Macintosh interface using a similar design of scrollbars and buttons, (see Appendix H). In designing Prototype 3.0, I tried to made the interface as intuitive as possible. I do not believe that the user should have to think about how to navigate through a program, it should be intuitive, and thus I incorporated this type of thinking into the design of Prototype 3.0 by keeping the interface simple.

Interactive Media is a new term meaning a dialog between two forms of intelligence. Some people will dispute the fact that the computer is not a form of intelligence. In essence the user is interacting with the programmer, designer, historian, and anyone else who had a hand in creating the actual computer program.

In designing an interactive program, programmers and designers try to anticipate every possible question that the user could possibly wonder about, pertaining to the subject matter of the program. In the case of Prototype 3.0 of the Graphic Design Archive, I tried to determine what the user would be interested in knowing as he/she sits in front of the computer screen. To do this I first had to identify who the user was; there will be more about this in section *Prototype 3* in this report.

The user should be able to use several tools to accomplish his/her goal. The important thing to remember is that the user must be able to have a dialog with the computer, to actually interact. This means that the user must walk away having learned something, no matter how small or insignificant. As I was working on this thesis, I had to ask myself at some point whether or not Interactive Media was the correct way to invoke this type of dialog of learning. The answer which I found was yes.

Interactive media is being used in many industries today for learning and research. Companies are finding that this type of learning is reducing their training time by large margins. Part of the reason is that the user can make his/her own choices about what they want to do. Subsequently, the user can progress

at his/her own speed, with a feeling of control over the learning process. Because of this factor, the user seldom becomes bored, thus learning increases.

As a research or archiving tool, the computer lends itself beautifully for storing and dispensing large amounts of data, whether it be text or imagery. In addition to the computer, the laser disc also lends itself quite nicely to storing large amounts of information. In the case of the Graphic Design Archive, this information is usually in the form of imagery. Able to perform several million instructions per second, the computer becomes useful when retrieving one image with a set of characteristics from a database of thousands of images.

The United States Military is using interactive media. They are starting to use it not only for teaching new personnel, but also for maintenance purposes. By this I mean that they are experimenting with putting interactive laser disc programs in such devices as tanks. These devices are becoming useful at times when there is a malfunction or breakdown. One of the tank crew can start up the program which will take them step by step, by first identifying the problem and then solving it. Because laser discs can also have motion video, a mechanic appears on the video monitor and show the soldier step by step how to repair the malfunction. It is not out of the question to see this type of technology coming to our homes, or even our own automobiles in the near future.

Laser Video Technology and it's Role in the Interactive World

Video technology is still in its infancy stages. Yet it has come a long way in the past few years. Historically, motion pictures were captured on long rolls of gelatin film (motion picture film). This process was fairly tedious, and the results could not be readily viewed. Finally video tape was developed to a state where it could finally replace motion picture film as the de facto medium.

There are many different types of video tapes. The one most commonly used today professionally is 1" video. This delivers the best quality in an analog format. Analog means that the recorded information is in wave form, similar to a sound wave. Video discs are recorded with these analog video signals. The advantage that a video disc has over a video tape recording is that it is permanent. This means that no matter how many times you play the video disc, it will never change in picture quality.

With video tape this is not the case, for every time the tape is played, the recorded information deteriorates. The reason for this difference is that the video tape moves across a play back head in the video tape player. This head actually comes into contact with the tape, thus causing friction, and deterioration. The laser disc however, never comes in contact with any playback part of the disc player. Instead the disc player uses an argon laser beam which directs a light ray onto the surface of the disc. The disc, because of it's reflective surface, causes the beam of light to bounce back into the lens of the player. This information is then reassembled as video wave forms and displayed on the video monitor.

Another advantage of a laser disc over any type of tape medium, is that the laser disc can access any frame on the disc in a matters of seconds. However there are several advantages which tapes have over disc, the primary one being that they can be recorded over many times. This makes them an ideal medium for use in the home, and might possibly be the reason why laser discs have never penetrated the home video market. Yet for archival purposes such as with the GDA, the permanent nature of the laser disc, along with its ability to randomly access frames makes it the perfect medium.

The video world is changing however, and the Archive is prepared to change with it. There are many new formats of video that promise much higher resolutions and archival methods. Some are already in use in other parts of the world. One such method is high definition television (HDTV). This was first developed in Japan and is slowly spreading to other parts of the world. Unfortunately, the United States is the slowest to change in this. However progress is being made here on our home soil.

HDTV is a wide screen television signal with much higher resolution. There are several different versions of HDTV in the foreign market right now. The U.S. Federal Department of Communications is testing to see which one would be the best to implement here in the U.S. It is greatly hoped that some day the Graphic Design Archive could be moved from regular NTSC video to HDTV format. To ensure that this move could someday be made, two careful measures were taken in the production of the archive.

Laser Video Technology and it's Role in the Interactive World

The first step in the archiving process was to capture the archive in its entirety on the medium which offered the best possible resolution. This proved to be 35mm motion picture film. Once this was accomplished, the archive was then recorded onto 1" composite video tape, which keeps the video signal separated into its most basic components. When the archive is someday ready to be moved to a better display medium, the 1" composite video tape will be used as the source. Secondly, as an additional precaution, the archive will be kept in its entirety on 35mm motion picture film.

In the case with the current archive however, the 1" video tape was then sent to a laser disc mastering plant, where the first test disc was pressed. This test disc was then inspected very closely before the approval was sent to produce the disc in large quantities. This current version of the disc will probably outlast its technology. The Graphic Design Archive will hopefully develop right along with the changes in the video and archival technology, thus utilizing faster accessing and much higher resolution.

Prototype 2.0

There have been several versions of the Graphic Design Archive. In this chapter I will deal only with Prototype 2.0. This is where my thesis began, and is subsequently the main source for many of the features in Prototype 3.0. Prototype 2.0 was created by Cathleen Britt, in 1989 as a thesis project. Cathy carried many good elements of prototype 1.0 over to her version. In addition she refined many of the elements, while adding some of her own. I volunteered on the the Graphic Design Archive development committee last year while Prototype 2.0 was under development. This gave me a valuable chance to learn all of the workings of the archive as they were been developed and refined.

The purpose of Prototype 2.0 was quite different than that of Prototype 3.0, as well as the imagery contained on the disc. The main purpose of Prototype 2.0, and Prototype 1.0 for that matter was to document the history of graphic design as a whole, or at least as a fairly large sample of the whole. With Prototype 2.0 there were 806 images on laser disc by some twenty pioneering graphic designers covering a period from 1930 to 1955. The amount of information represented on the data cards was minimal. The data cards consisted of only a few categories including the title, designer, date, and comments. Yet for the purpose of the prototype, this seemed to be sufficient.

Prototype 2.0 was created for a museum installation with the audience ranging from a scholar of graphic design to a casual browser. This I believe was one of the first major problems of this prototype; the audience was too wide. The program had to assume many things about the user and then account for them. For instance if the user knew nothing about graphic design, the program would

Prototype 2.0

have to be simple enough so the user would not get frustrated and walk away. In the same respect, if the user was very experienced with both graphic design and computers, the program had to be challenging enough to captivate his/her interest, so he/she would not get bored with the program.

Prototype 2.0 was created in Hypercard® version 1.2, an Apple Computer, Inc. software development program. This works on an index card metaphor, where the usable screen is limited to 512 x 342 in one bit (black and white) mode. This was another part of the project which I felt needed improving upon and thus addressed in Prototype 3.0.

Life of Lester Beall and the Beall Collection

Lester Thomas Beall was born in Kansas City, Missouri, on March 14th, 1903. He died in New York City, on June 20th, 1969. Lester Beall was a pioneer of American graphic design, who was admired by his contemporaries. As a youth, Beall grew up in both St. Louis and Chicago. In 1927, Beall began his career after graduating from Chicago's Lane Technical School. In 1928, he was married to Dorothy Wells Miller, and fathered two children. His son Lester Beall, Jr., was born in 1928, while his daughter was born in 1935.

Initially Beall was unsure about what he wanted to do as far as a career. He worked as an illustrator for a while before he realized he wanted to do something related to graphics. The term, graphic design had only begun to emerge. After talking to a friend who had been trained at the Bauhaus in Germany, Beall decided that he would pursue a career in graphic design. By 1935, Beall moved to New York City, and by September of that year, opened his own business on Manhattan's East Side. Beall's business was very successful, and by 1936, he moved his home to Wilton, Connecticut.

By 1946 Lester Beall had moved his New York office to 580 Fifth Avenue, New York. He was still running his business in New York, when he opened another office at his new home at Dumbarton Farm in Brookfield Center, Connecticut in 1950. The business was being run by Lester Beall from both his New York office and Dumbarton Farm. But by 1955, Beall consolidated all of his business operations to his rural Connecticut office at Dumbarton Farm. Beall felt that a designer's environment was very important to him, thus justifying the move to Connecticut.

Life of Lester Beall and the Beall Collection

Beall had some very prominent clients such as Aluminum Company of America, Caterpillar Tractor, International Paper Co., Martin Marietta, and Rohm and Haas. The projects which Beall undertook for these companies ranged from brochures to corporate identities. Many of his designs were considered extremely innovative for the time. He was looked upon very highly in the design profession, and was a popular lecturer.

The Lester Beall Collection at RIT is quite extensive. The material, which is housed in archival boxes, ranges from hotel receipts to corporate identity projects. The work contained in the archive represent the bulk of Mr. Beall's professional career. Personal notes are even filed in special categories. Design projects are represented from sketches to final printed pieces. There is correspondence between Beall and various clients. There is also a large section of photographs, including a number of Dumbarton Farm in Connecticut. This collection was maintained by his wife Mrs. Dorothy Miller Beall. After her death, the Beall family donated the collection to RIT.

In early September of 1989, Roger Remington and I determined we might use just the imagery of Lester Beall's archive for Prototype 3.0. We did not want to lose sight however of the Graphic Design Archive as a whole. This is when I first began thinking of a node system. A node system is a logical breakdown of a greater entity into smaller more manageable pieces. As I developed this node system, I noticed some similarities to fractals and hyper-networks. The posters which I developed for the thesis show, depict the developmental structure of fractals and hyper-networks, and show how some of these properties relate to the GDA, (see Appendix A).

The GDA can be related to a node system in the following way. Assuming the GDA to be the main node of the program, several smaller nodes are attached to it. In turn each of these smaller nodes has its accompanying smaller node colony, this can continue for several steps. Each smaller node retains a similar structure as the one before it. Each node attached to the main GDA node, represents a graphic designer.

Prototype 3.0, which is based on Lester Beall as the main designer, is one of these such nodes. In time there can be several other nodes like Prototype 3.0, all of which will be based on the GDA. These other nodes will contain other designers, such as Paul Rand, Alvin Lustig, or Herb Lubalin.

Like Prototype 3.0, all of the designer nodes will have various levels of information, and in turn these will also have deeper levels. With Prototype 3.0, the user can traverse down several different levels without ever feeling like he/she is in a

different program. This is because of the consistency of design from one level to another. Yet because of the navigational tools the user never becomes disoriented of where he/she is in the program.

Prototype 3.0 allows the user to make many different decisions. This is called hyper-media. It is based on the way that the human brain functions. Because of the neuron structure of the brain, which is not linear, the human brain often jumps from one thought to another. Hyper software is an attempt to capture that essence. It allows the user to move freely at his/her own pace. The user is free to make many decisions at any point in the program. This is a vast difference from linear material. A text book a good example of linear structure. While reading a book the reader generally reads from the front of the book to the back. If the reader decides that he/she would like to skip around in the book a little, the result can often be confusion.

I looked at many different structures of networks, from binary to neurons. The one I felt represented what I was trying to accomplish in Prototype 3.0, is depicted in Poster #2, (see Appendix A). This is called a hyper network because of the fact that at any point, the user can jump to any other point and then back again without losing track of where he/she is. This network structure seemed to fit in quite nicely with the node structure of the archive, and is brought together in Poster 3, (see Appendix A).

Prototype 3.0

Prototype 3.0 of the Graphic Design Archive is comprised of images from the Lester Beall Archive. The database is made up of some 5,000 images of his work. There have been nearly 4,000 details constructed to show close ups of individual items in the collection. In many cases the details are so clear that even the small body text on advertisements can be read. This is a vast improvement over past prototypes of the archive and is attributed a careful benching process of the Beall Collection on to high resolution film

This new disc contains just over 29,000 images of graphic design history, the largest section being the Lester Beall Collection. For each item in the Collection there is a data card which holds information about that item. During the process of data entry, each physical data card was taken and carefully entered into a custom "Data Cards" computer program, which I designed solely for that purpose. When a sufficient amount of data was entered, I transferred that data to Prototype 3.0. Currently, there are 244 items of the Beall Collection represented in Prototype 3.0.

After determining who the main focus of Prototype 3.0 was to be, I was then faced with the problem of identifying an audience to direct the project toward. After much deliberation, Roger Remington and I came to the conclusion that the main client for Prototype 3.0 would be the Wallace Memorial Library at RIT. The audience would therefore be librarians, archivists, historians, researchers, and students. The main purpose for the prototype would be archival. The Wallace Memorial Library will be acquiring the Beall Collection from the Graphic Design Department sometime during the summer of 1990.

Prototype 3.0

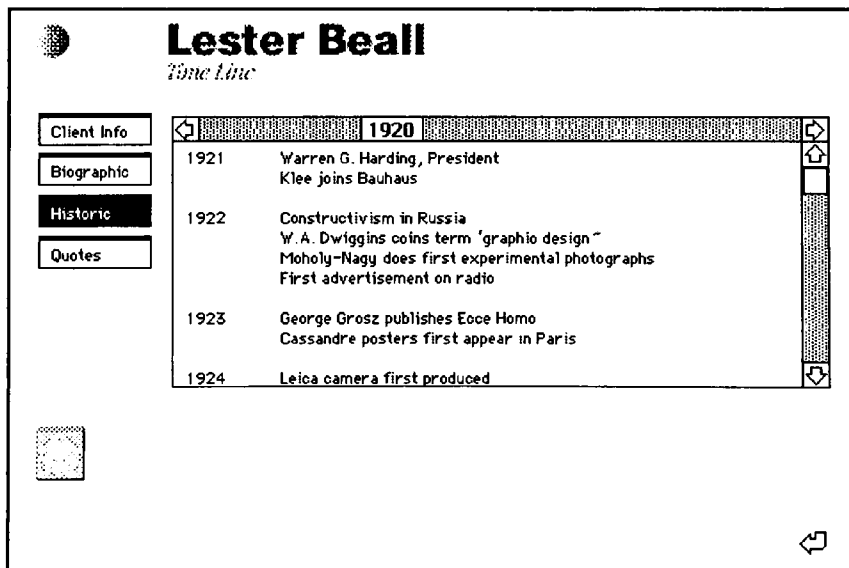
With the main audience specified, I was then able to begin to think about what tools would be appropriate to suit the user. The first thing I did was to go back to Prototype 2.0 and look at some of the tools there. The new prototype is based on only one designer, while the previous one was based on many. Because of this there were several things I was not able to use. One of these was the taxonomy or tree structure of the previous version. However there was quite a bit that was usable in the new version, although slight alterations were required to each.

During several of our GDA meetings, ideas were tossed around for new tools of Prototype 3.0. Among these was the need for some kind of timeline. This required much deliberation on my part. I wanted to come up with a timeline that would have some unique features, while keeping in mind that it had to be very easy to use. I began work on the timeline around the beginning of January of 1990. I played around with many ideas (see Appendix D) trying to make the interface graphical. However I found that the interface was clunky to say the least. Then after several meetings with Roger about this, I decided to try a timeline that be able to do some unique things.

This new timeline is able to scroll across the top from left to right displaying decades, and from top to bottom displaying individual years. The most unique thing about this new timeline though, was its ability to cross reference information. This was achieved by having four different sections, each with its own series of decades, and individual years. The first and most important section of the timeline was the client information. This was taken directly from the master

lists of the Collection which RIT received from the Beall family. This information was broken down chronologically, into decades. When the user chooses the decade, he/she is able to scroll down the years with the use of a scroll bar. From this point the user can cross reference the date he/she chose to any of the three other sections.

These other sections include, a timeline on Lester Beall's personal life, a section on historical events of the world during the time when Lester Beall was active, and finally, an entire section on Lester Beall's quotes. These quotes were carefully selected by Roger through a very long process, of sorting. The result is a rich cross section of select quotes by Lester Beall, totaling some 408 records.



Prototype 3.0

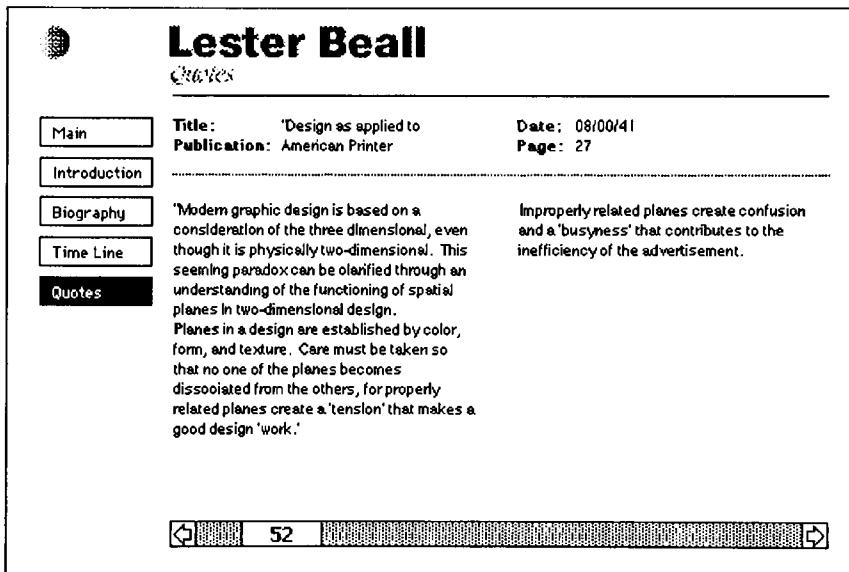
The timeline was just one of the tools which the GDA committee decided on. Probably the most important tool is the searching section of the program. Although most of the search routines were taken from Prototype 2.0, the interface was redesigned in Prototype 3.0. The categories on which one might search were also changed. Topics like "Location" and "Designer" were no longer needed, since all of the images were created in the United States, and the designer was Lester Beall.

I consolidated the way in which the searches are conducted. In Prototype 2.0 there were two kinds of searches, single searches and query searches. I thought that this was a bit confusing, and so did away with the two types making them one. Now the user doesn't have to be bothered with which search method he/she is in because the program adjusts itself accordingly.

The screenshot shows a search interface for the Lester Beall Graphic Design Archive. It features a title bar with a logo and the text "Lester Beall" and "Graphic Design Archive". Below the title bar is a search form with several input fields and buttons. The form is organized into a table-like structure with labels on the left and input fields on the right. The input fields are: Title (empty), Client (Abbott Laboratories), Medium (Print), Subject (empty), Box# (empty), and File (empty). The Client, Medium, and Subject fields have small numeric indicators to their right: 4, 93, and an empty field respectively. At the bottom of the form are three buttons: "Cancel", "Query Search", and "Remove". A "Quit" button is located at the bottom left of the window.

Lester Beall	
<i>Graphic Design Archive</i>	
Title	<input type="text"/>
Client	<input type="text" value="Abbott Laboratories"/> 4
Medium	<input type="text" value="Print"/> 93
Subject	<input type="text"/>
Box#	<input type="text"/>
File	<input type="text"/>
<input type="button" value="Cancel"/>	<input type="button" value="Query Search"/> <input type="button" value="Remove"/>
<input type="button" value="Quit"/>	

The quotes which Roger compiled were a unique addition to the project. However, I was not happy because the only way in which the user could access this information was through the Timeline section. I made another section where the user could browse through all of the quotes with a simple scrollbar. The scrollbar moves horizontally displaying individual cards which contain quotes, where the quote was taken from, the date of the quote, and the source of the quote.





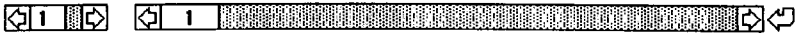
The screenshot shows a web interface for Lester Beall. At the top left is a circular logo. The main heading is "Lester Beall" with the word "quotes" in a smaller font below it. On the left side, there is a vertical navigation menu with buttons for "Main", "Introduction", "Biography", "Time Line", and "Quotes". The "Quotes" button is highlighted in black. To the right of the menu, there is a section for a quote. It includes a "Title:" field with the text "Design as applied to", a "Publication:" field with "American Printer", a "Date:" field with "08/00/41", and a "Page:" field with "27". Below this, there are two columns of text. The left column contains a quote: "Modern graphic design is based on a consideration of the three dimensional, even though it is physically two-dimensional. This seeming paradox can be clarified through an understanding of the functioning of spatial planes in two-dimensional design. Planes in a design are established by color, form, and texture. Care must be taken so that no one of the planes becomes dissociated from the others, for properly related planes create a 'tension' that makes a good design 'work.'" The right column contains a shorter quote: "Improperly related planes create confusion and a 'busyness' that contributes to the inefficiency of the advertisement." At the bottom of the interface is a horizontal scrollbar with the number "52" in the center.

The user can also browse through a short biography of Lester Beall, and the Introduction section which contains a brief history of the GDA. The biography section contains information about Beall's early years as a young designer in Chicago, New York, and Connecticut. The introduction section talks about where the GDA has been, and where it hopes to be in future years.

The final tool in Prototype 3.0 is the Data Cards. These were totally redesigned for the new prototype. The new cards hold much more information than previous versions, making the GDA project much richer, (see Appendix G). The addition of details in this version of the GDA posed some problems initially.

I had to come up with an easy solution so the user would be able to know how many details, if any, there were, and how to browse through them. I accomplished this by letting the user select the small scroll bar under the word "Details" and moves it left and right. By doing this the user can browse through details of images that are be displayed on a separate monitor. Some data cards do not have any details. In these circumstances the program again adjusts itself by hiding the "Details" scroll bar.

		Lester Beall	
		<i>Data Cards</i>	
Date	1936	Location	Box 1
Title	ABD Vitamin Capsules Folder		
Art Director			
Artist	Beall, Lester		
Client	Abbott Laboratories		

Description	Width 12"	Height 12"	Depth 0"
Size (inches)	Print		
Medium	Brochure		
Subject	-----		
Copyright			Date
Publisher	Beall Collection		Date
Source			
Comments			
			
Details			
			

Prototype 3.0

The only real limitation which I was faced with during the course of this project was that Hypercard® can only display black and white (1 bit). I was not happy with this, for I wanted to show images of Beall in the biographical section, on the Macintosh screen in addition to the separate monitor being driven by the laser disc player. To archive this I used an external command called ShowPict. This allowed me to be able to display 256 levels of grays or colors. I decided that if I was going to display anything on the Macintosh Monitor, it was going to be in levels of gray, as not to distract from the video monitor.

I used the Apple Scanner® to scan in photographs of Beall, and some biographical shots. I used Adobe Photoshop® by Adobe™ and Letrastudio® by Letraset™, to create anti-aliased type and retouch the photographs. This allowed me to display the major titles, with type that appeared to be smooth. It is through such innovations as this, as well as unique tools and a much richer amount of data and imagery which make Prototype 3.0 possibly the best version of the Graphic Design Archive yet.

Conclusion

Someday, I hope to see the Graphic Design Archive being promoted and supported on an international level. Interactive media is growing by leaps and bounds every day. In a short time, the technology will be present to broadcast video through the interactive control of a home user. This means that anyone who can watch the television and operate a computer, will be able to interact with the GDA and programs like it. The potential is there to have an entire museum represented on video right in the comfort of one's own family room.

We are not far away from this now, but currently this idea is still just that, an idea waiting for the technology to catch up. With the speed with which technology is growing though, it would not be out of the question to see new types of software being developed and implemented, which utilize these new ideas, in the next few years. *ABC News Interactive*, a division of ABC, is developing new programs which take the user in this new direction of interactive control. To date *ABC News Interactive* have created three such programs with laser video discs.

As the GDA gains support, I hope that other schools will begin to join the consortium of collections which the GDA has begun.

Sources Consulted

Apple Computer, Inc. Human Interface Guidelines. The Apple Desktop Interface.

Reading, Massachusetts: Addison-Wesley Publishing Co., Inc., 1987

Goodman, Danny. The Complete Hypercard Handbook, Toronto, New York,

London, Sydney, Auckland: Bantam Books, 1987.

Remington, R. Roger and Hodik, Barbara J. "Lester Beall: A Look Back,"

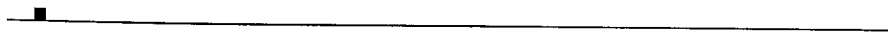
Communication Arts, September/October 1985.

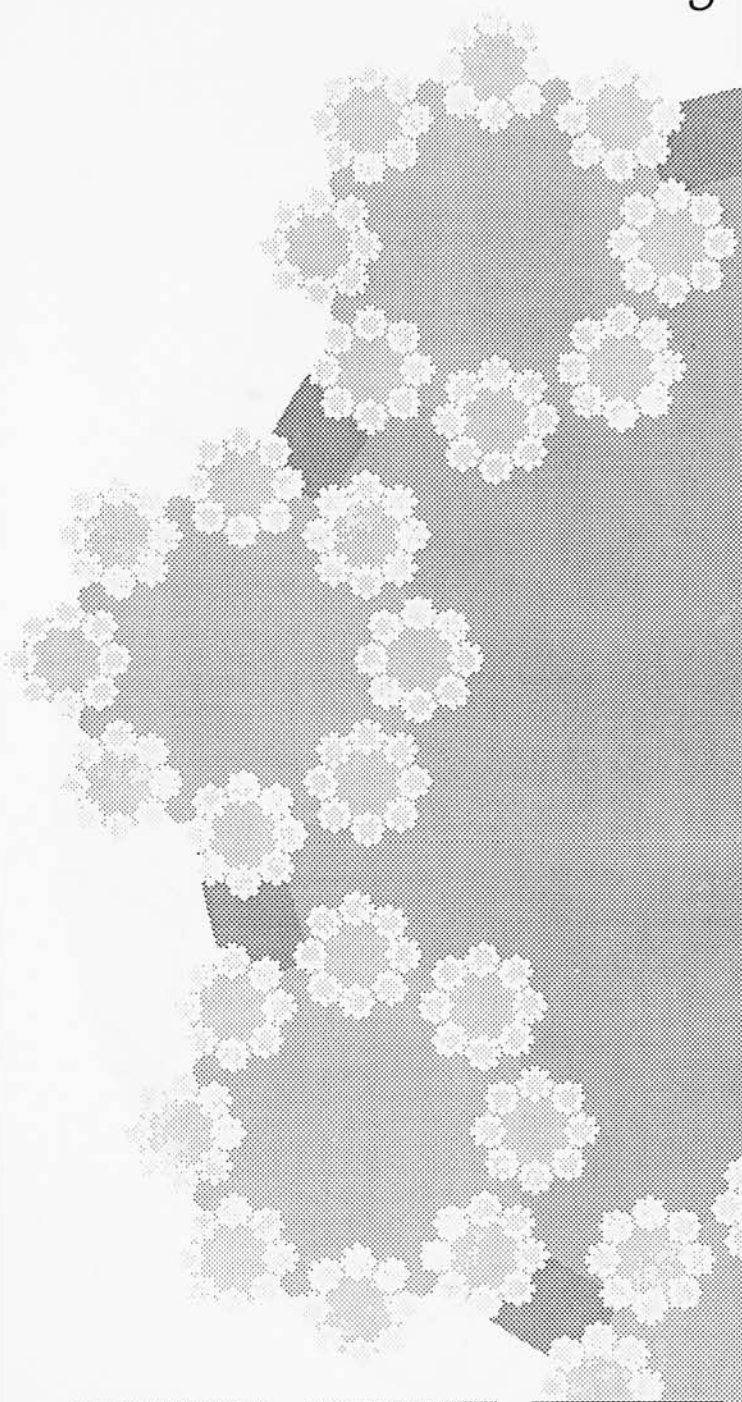
Apple Computer, Inc. The Glenn Gould Profile. Minds in Motion. Fall 1989.

Britt, Cathleen. "A Guide for Interactive Interface Design," Thesis Report,

Rochester Institute of Technology Archives, 1989.

Posters





Fractals are mathematically occurring shapes which go on for infinity. This is the underlying structure for the Graphic Design Archive. The fractal illustrated above has certain properties. One being that it is always symmetrical. The second property is that the sub-section or nodes are always a duplicate structure of the parent shape.

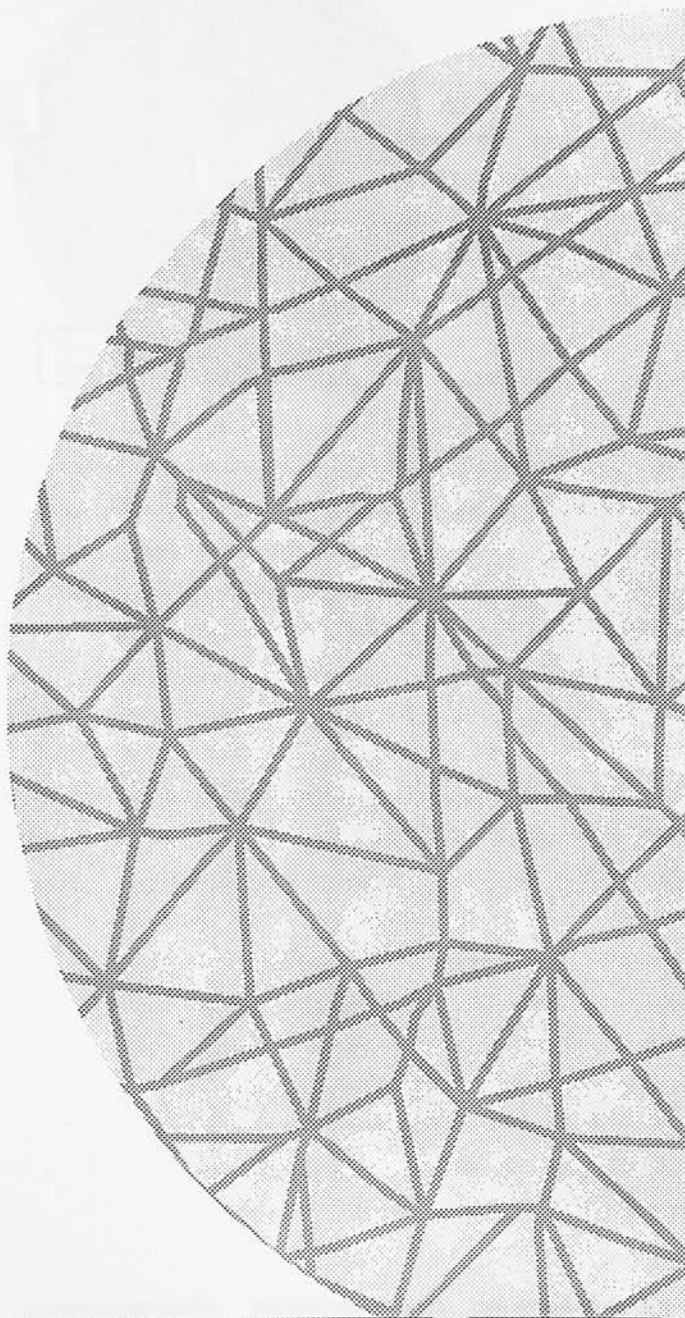
This prototype of the Graphic Design Archive, represents one subsection of the archive as a whole entity. As

with fractals this subsection of the main archive has several subsections of its own. In turn each one of these subsections also has several of its own, and so this refinement process continues. Of course there is no end to the process.

When looking at a fractal one may see chaos, yet in all chaos there is structure. In this case the structure is very evident. This particular fractal is based on refinement. The refinement continues for infinity. A viewer can zoom into a fractal,

or enter down another level. Each time a new layer is revealed so it reveals information, yet it is displayed in a similar structure to the prior layer.

The Graphic Design Archive, being that it is based on a naturally occurring structure has a very strong structure of its own. As more and more refinements are made to the archive, such as the addition of the Laser Beat Archive and this prototype, so shall the Graphic Design Archive continue to grow.



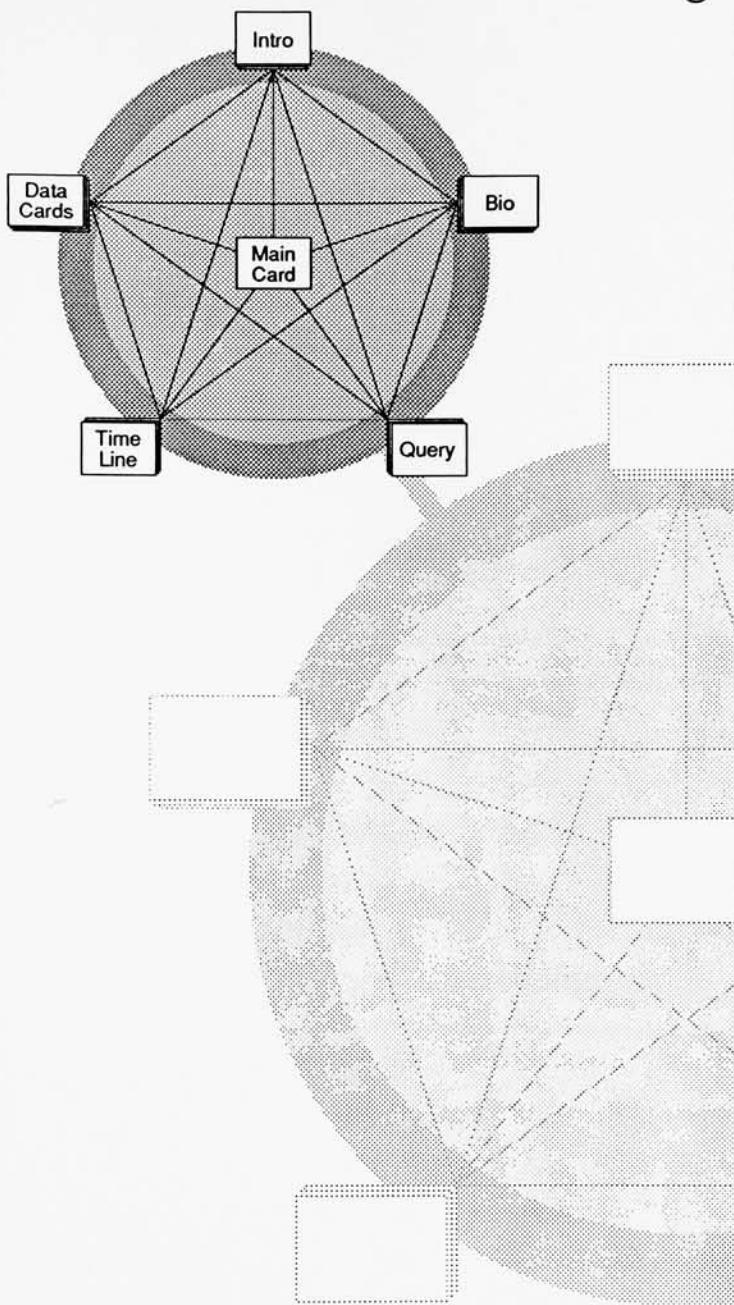
Hyper media is a fairly new term, basically it is any type of media that tries to structurally resemble the way that the human brain functions. When the brain operates it has the tendency to jump from one topic to another. This type of thought is called "hyper-thought". Hyper is not necessarily a bad thing, as in the term "hyper-active". Instead hyper implies the jumping from one thought or action at any time.

In this sense, hyper media works much

the same way as the human brain and thus the learning curve for using this type of medium, as opposed to traditional methods is much better.

When using a hyper media program, the user is free to jump to as many different topics as he/she wishes to. A truly hyper program can jump from one point to any other point in the program. With the use of the laser disc, this become much more interesting, for the number of choices reaches tens of thousands.

A laser disc has the ability to hold up to 54,000 images on the side. The computer in turn can jump to any image in any order the user wishes. It is by this nature that the Graphic Design Archive was begun. The new disc, holds an excess of 30,000 images. For this prototype there are about 5,000 images of Lester Beall's, an early American graphic designer, work. There is also some 4,000 additional images which are details of Mr. Beall's work, which can be accessed.



The theory behind this prototype to the Graphic Design Archive is to use a node system in the creation of one small archive to fit as a sprocket into a larger archive. This main archive would be the central processor for all of the nodes.

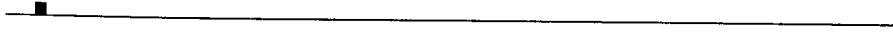
This thesis represents one of these nodes. The designer featured in this archive is Lester Beall. This is the third prototype in the creation of the main Graphic Design Archive.

A node is a smaller subset of a much larger entity. In this case, Prototype 3 is only one very small element of the whole. When this and many other nodes are completed, then an archive of immense information will be at the finger tips of whomever uses this.

The main use of Prototype 3 will be both as a research tool of Lester Beall's work, as well as a finding guide for the actual pieces, which will be housed in the Wallace Memorial Library at R.I.T.

Accompanying each data card of information in this archive is an image which is displayed on a separate monitor. This image is stored on an optical laser disc. Each disc has the ability to hold up to 54,000 still images per side. When combined with the speed of the computer, a researcher can look at thousands of pieces of work, along with accompanying data in very little time. The Graphic Design Archive is but just one of the many uses for laser discs and computers.

Scripting



SCRIPTS FOR STACK: Beall Archive

```
-- STACK SCRIPT -----
on openStack
  titlebar hIde
  hide menuBar
  hide msg
  -- set userLevel to 1
  global initialized
  if initialized is true then
    exit openStack
  else
    --Initialized directs hypercard to this stack (from the card stack)
    --(or from the reference stack) on first opening.
    --This insures that all necessary global variables are declared.
    --The user should always start from this stack.

    --Install these handlers into each interactive video stack.
    --These globals must be initially declared or video won't work!

    global SPortGlobals --serial port stuff
    global typeOfVideo,lastVideoFrame,blankNextVideo,videoSpeed,videoMode
    put true into initialized
    setVideoPlayer "Sony 1000"
  end if
end openStack

on doMenu command
  if command = "quit hyperCard" then
    put empty into cd fld "theStack" of cd "Main"
    put empty into cd fld "Images" of cd "Main"
    repeat with x = 1 to 5
      put empty into cd fld x of cd "Main"
      put empty into cd fld (x+5) of cd "Main"
      put "no entry" into cd field ("list"&x) of cd "Main"
    end repeat
    put empty into cd fld itemHolder of cd "Main"
    repeat with x = 27 to 28
      hide card button x of cd "Main"
    end repeat
    hide cd btn "Data Cards" of cd "Main"
    hide cd field theStack of cd "Main"
    hide cd field images of cd "Main"
    put empty into card field "itemHolder" of cd "Main"
    send mouseUp to cd btn "Cancel" of cd "Main"
  end if
  pass domenu
end domenu

function ClickLine
  -- Script written by Luc Perron in the spring of 1988 at RIT
  -- This function returns the line number of where the mouse has been
  -- clicked in a field (Scrolling or not).
  -- Ex: put ClickLine() into LineNumber
  put item 2 of the rect of the Target into Top
  put item 2 of the clickLoc into Y
```

```

if the style of the Target is "scrolling" then
  return (Y - Top + scroll of the Target) div ~
  (textHeight of the Target) + 1
else
  return (Y - Top) div (textHeight of the Target) + 1
end if
end ClickLine

```

```

—This function returns the union of the two lists.
—This script is the result of the work of the spring 1988
—MicroComputer Control class taught by Steve Kurtz at RIT
function PlusList listOne, listTwo
  repeat with counter = 1 to number of lines in listOne
    if line counter of listOne is not in listTwo then
      put return & line counter of listOne after listTwo
    end if
  end repeat
  return listTwo
end PlusList

```

```

—This function finds the intersection of the 2 lists
—This script is the result of the work of the spring 1988
—MicroComputer Control class taught by Steve Kurtz at RIT
function InterList listOne, listTwo
  put empty into difference
  repeat with counter = 1 to number of lines in listOne
    if ((line counter of listOne)&return) is in listTwo then
      put line counter of listOne & return after difference
    end if
  end repeat
  if last character of difference is return then
    delete last character of difference
  end if
  return difference
end InterList

```

```

on sort
  — All cards in the three GDA stacks are in a specific order!
  answer "it would not be wise to sort this stack" with "OK"
  exit sort
end sort

```

```

** BACKGROUND #1 *****
on opencard
  if the short name of this cd <> "Main" then
    show bg btn "Next"
  else hide bg btn "Next"
  global counter
  put 0 into counter
end opencard

```

```

on idle
  global counter
  put the mousetloc into mouseCheck
  wait 6 ticks
  if the mousetloc <> mouseCheck then
    put 0 into counter
  else

```

```

    put counter + 1 into counter
  end if
  if counter = 3000 then
    put 0 into counter
    go to cd 1 of stack "Beall Archive"
  end if
end idle
** BKGND!#1, BUTTON #1: Introduction *****
on mouseUp
  go to cd "Intro"
end mouseUp

** BKGND!#1, BUTTON #2: Time Line *****
on mouseUp
  push this cd
  go to stack "Time Line"
end mouseUp

** BKGND!#1, BUTTON #3: Next *****
on mouseUp
  go next
end mouseUp

** BKGND!#1, BUTTON #4: Next *****
on mouseUp
  go prev
end mouseUp

** BKGND!#1, BUTTON #5: Biography *****
on mouseUp
  go to cd "Bio"
end mouseUp

** BKGND!#1, BUTTON #6: Main *****
on mouseUp
  go to cd "Main"
end mouseUp

** BKGND!#1, BUTTON #7: Quotes *****
on mouseUp
  go to stack "Quotes"
end mouseUp

** BKGND!#1, BUTTON #8 *****
on mouseUp
  go to cd 1
end mouseUp

** BACKGROUND #2 *****
on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
end idle

```

```

on opencard
  set the scroll of bg fld 2 to 0
  set the scroll of bg fld 3 to 0
end opencard
** BKGND:#2, BUTTON #1: Content Count *****
on mouseUp
  put the number of lines in field contentList
  put " " after message
  put the number of items in field IDlist after message
end mouseUp

** CARD #1: Start *****
on opencard
  set cursor to Hand
  showpict "GDA LOGO", 20,19
  showpict "GDA-p3", 194,12
  showpict "Lester Beall-12pt", 393,135 —82
  showpict "Big Beall", 203,60
  playVideo firstFrame,lastFrame
  repeat until the mcuseclick
    showPict "Start", 393,285
    wait 50 ticks
    showPict "Blank", 393,285
    wait 25 ticks
  end repeat
  searchVideo 1
  go next
end opencard

** CARD #2: Main *****
—The "Quit" button is hidden
—The button "Hidden" is also hidden—this shows hidden fields

on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "GDA", 95,32
  if cd fld theStack is empty then hide cd picture
  else
    show cd picture
  end if
  pass idle
end idle

on reset
  — Reset card buttons and fields to "Stack Builder" state
  hide cd field theSelection of cd "Main"
  show cd button queryCover of cd "Main"
  hide cd btn "Cover" of cd "Main"
  hide cd btn "Block6" of cd "Main"
  hide cd btn "File" of cd "Main"
  hide cd btn "Block5" of cd "Main"
  hide cd btn "Box#" of cd "Main"
  hide cd btn "Block4" of cd "Main"
  hide cd btn "Subject" of cd "Main"
  hide cd btn "Block3" of cd "Main"
  hide cd btn "Medium" of cd "Main"

```

```

hide cd btn "Block2" of cd "Main"
hide cd btn "Client" of cd "Main"
hide cd btn "Block1" of cd "Main"
hide cd btn "Title" of cd "Main"
set hilite of target to false
show cd btn "Search" of cd "Main"

```

```

repeat with n = 5 down to 1
  put n into temp
  add 5 to temp
  hide cd field temp of cd "Main"
  hide cd field n of cd "Main"
  hide cd button n of cd "Main"
  set hilite of cd button n of cd "Main" to false
end repeat
hide cd btn "Cancel" of cd "Main"
hide cd button "query Search" of cd "Main"
hide cd button "remove" of cd "Main"
hide card button "add to stack" of cd "Main"
put empty into card field result of cd "Main"
hide card field result of cd "Main"
if (cd field theStack of cd "Main") is not empty then
  show cd field theStack
  show cd field images
  repeat with z = 27 to 28
    show cd button z
  end repeat
  show cd btn "Data Cards"
end if
end reset

```

```

** CARD #2, FIELD #12: theSelection *****
--Lists of card ids and frame numbers are stored on cards.
--Each list is sorted a different way.
--DesignerList is sorted by Date, Taxonomy, Medium, Location, Designer
--CardName is the name of the card and list from which the item came.

--ItemHolder is a hidden field that contains thecard ids
--and frame numbers of everything in the Stack.
--The first word of each line is a card id.
--The second word of each line is its corresponding frame number.

--Place is either "stackBuilder" or "query"
--This makes sure that the info.
--gets put into the correct containers (hidden fields).

```

```

on mouseUp
  global cardName

  if number of lines in cd field theStack = 12 then
    answer "The Stack is too full, remove something first" with "OK"
    exit mouseUp
  end if

  put clickLine () into lineNum
  select line lineNum of card field theSelection
  set cursor to 4
  put (item lineNum of bg field idList of cd cardName) into tempitem

```

—tempitem is a temporary container.

if first line of tempitem is empty then delete first line of tempitem
put number of lines in tempitem into imageHolder
—imageHolder is a temporary container.

```
hide card field "theSelection"  
set hilite of cd button cardName to false  
put (line lineNum of card field theSelection)-  
into cd field cardName  
if cardName is "Title" then  
  put tempitem into cd field "list1"  
  put imageHolder into cd fld ImageFld1  
else if cardName is "Client" then  
  put tempitem into cd field "list2"  
  put imageHolder into cd fld ImageFld2  
else if cardName is "Medium" then  
  put tempitem into cd field "list3"  
  put imageHolder into cd fld ImageFld3  
else if cardName is "Subject" then  
  put tempitem into cd field "list4"  
  put imageHolder into cd fld ImageFld4  
else if cardName is "Box#" then  
  put tempitem into cd field "list5"  
  put imageHolder into cd fld ImageFld5  
end if  
show cd btn "Query Search"  
show cd btn "Remove"  
end mouseUp
```

```
** CARD #2, FIELD #13: itemHolder *****  
on mouseUp
```

```
end mouseUp
```

```
** CARD #2, FIELD #19: theStack *****  
on mouseUp  
global indexHolder, counter, theNum, source  
put empty into source  
put empty into indexHolder  
put empty into counter  
put clickLine () into theNum
```

— IndexHolder keeps track of the number of the video image
— that is on the screen when the user has stopped scanning.
— It allows the user to scan images, stop on a particular one
— and go to the correct data card for that image.
— It is important that the IndexHolder is reset
— every time a new line is selected
— so that the new item can be viewed in the same way.

— Counter keeps track of the line number
— of the item that is being scanned. The forward and backward
— scan buttons add and subtract from counter.

— theNum represents both the line in field "theStack"
— that has been selected and the number of the corresponding item
— in hidden card field "itemHolder."

```

set cursor to 4

put theTicks into originalTicks
— this script differentiates the single click operations
— from the double click operations. The user can double click
— anywhere in card field "theStack" to select the whole thing.
wait 15 ticks
if the mouseClick is true then —if double clicked...
  put 0 into temp
  repeat with x = 1 to number of lines in cd field theStack
    add 1 to temp
    put x into buttonNum
    add 14 to buttonNum
    set hilite of cd button buttonNum to true
    — Hilite all the lines that are not empty
    — visual feedback for "select all."
  end repeat
  if temp = 1 then
    put 1 into theNum
  else
    put "selectAll" into theNum
  end if

else
  if line theNum of cd field theStack is empty then
    —if the user clicks once on an empty line turn off all buttons.
    put empty into theNum
    repeat with x = 1 to number of lines in cd field theStack
      put x into buttonNum
      add 14 to buttonNum
      set hilite of cd button buttonNum to false
    end repeat
    exit mouseUp

  else
    put (theNum + 14) into oneButton
    repeat with x = 1 to number of lines in cd field theStack
      — If the user clicks once on a full line
      — hilite the correct button and turn the others off.
      put x into buttonNum
      add 14 to buttonNum
      set hilite of cd button buttonNum to false
    end repeat
    set hilite of cd button oneButton to true
  end if
end mouseUp
-- CARD -- 2, BUTTON #1 -----
on mouseUp
—Reset button hilites...
if hilite of target is true then
  set hilite of target to false
else
  repeat with x = 1 to 5
    set hilite of cd button x to false
  end repeat
  set hilite of target to true

```


end if
end mouseUp

```
"" CARD "" 2, BUTTON #2 ""  
on mouseUp  
  —Reset button hilites...  
  if hilite of target is true then  
    set hilite of target to false  
  else  
    repeat with x = 1 to 5  
      set hilite of cd button x to false  
    end repeat  
    set hilite of target to true  
  end if  
end mouseUp
```

```
"" CARD "" 2, BUTTON #3 ""  
on mouseUp  
  —Reset button hilites...  
  if hilite of target is true then  
    set hilite of target to false  
  else  
    repeat with x = 1 to 5  
      set hilite of cd button x to false  
    end repeat  
    set hilite of target to true  
  end if  
end mouseUp
```

```
"" CARD "" 2, BUTTON #4 ""  
on mouseUp  
  —Reset button hilites...  
  if hilite of target is true then  
    set hilite of target to false  
  else  
    repeat with x = 1 to 5  
      set hilite of cd button x to false  
    end repeat  
    set hilite of target to true  
  end if  
end mouseUp
```

```
"" CARD "" 2, BUTTON #5 ""  
on mouseUp  
  —Reset button hilites...  
  if hilite of target is true then  
    set hilite of target to false  
  else  
    repeat with x = 1 to 5  
      set hilite of cd button x to false  
    end repeat  
    set hilite of target to true  
  end if  
end mouseUp
```

```
"" CARD "" 2, BUTTON #6: Query Search ""  
— getFinalResult was written by Steve Kurtz
```

- It determines which of the 5 hidden lists will be used for the query
- and calls the `interList` function (in the script of this stack)

```

on getFinalResult
  set cursor to 4
  global finalResult, nextList
  put empty into finalResult
  put empty into lists

```

- check to see which of the lists contain card ids and frame numbers

```

repeat with N = 1 to 5
  if cd field ("list"&N) is not "no entry" then
    put (cd field ("list"&N)&"," after lists
  end if
end repeat
if number of items in lists =1 then
  put 0 into flag
  repeat with counter = 1 to 5
    if cd fld counter <> empty then
      put counter into flag
    end if
  end repeat
  put -1 into finalResult
  put cd fld flag & return after card field theStack
  put cd fld ("ImageFld"&flag) & return after card field images
  put cd fld ("list"&flag)&","&return after cd fld itemHolder
  show cd field theStack
  show cd field images
  repeat with x = 27 to 28
    show card button x
  end repeat
  show cd btn "Data Cards"

```

```

else
  show cd btn "Cover"
  put "dummy item" after lists
  put item 1 of lists into finalResult
  delete item 1 of lists

```

```

—if there are more than 2 constraints
—intersect the lists 2 at a time until there are no more
—put the result of the intersection into finalResult
repeat while number of items of lists > 1
  put item 1 of lists into nextList
  delete item 1 of lists
  put interList(finalResult, nextList) into finalResult
  —interList is a fcn in stack script
end repeat
end if
end getFinalResult

```

```

on mouseUp
  repeat with x = 1 to 5
    set hilite of cd button x to false
  end repeat

```

```

global finalResult
—finalResult is the end product of getFinalResult
—it holds the result of the constrained search of the database
set hilite of target to true
put empty into temp
set cursor to 4
repeat with x = 1 to 5
  set hilite of cd button x to false
  hide cd button x
  if cd field x is not empty then put x after temp
end repeat

—Can't conduct a search if there are no constraints!
if temp = empty then
  answer "Make some selections first!" with "OK"
  set hilite of target to false
  repeat with x = 1 to 5
    show cd button x
  end repeat
  show cd button remove
  hide cd btn "Cover"
  exit mouseUp
else
  hide cd button "remove"
  getFinalResult —See above
  if finalResult = -1 then
    put -1 into lineTotal
  else
    put number of lines in finalResult into lineTotal
  end if
  —lineTotal is the no. of images that match all constraints

—Let the user know what the result of the search is...
if lineTotal = 0 then
  answer "There are no matches for this query." with "OK"
  if it is "OK" then set hilite of target to false
  repeat with x = 1 to 5
    show cd button x
  end repeat
  show cd button "Remove"
  hide cd btn "Cover"
  exit mouseUp
else if lineTotal = -1 then
  set hilite of target to false
  reset
  exit mouseUp
else
  if lineTotal = 1 then
    put "The result is "&lineTotal&" image!" ↵
    into card field result
  else
    put "The result is "&lineTotal&" images!" ↵
    into card field result
  end if
end if

show card field result

```

```

    show card button "add to stack"
    set hilite of target to false
end if
end mouseUp

** CARD ** 2, BUTTON #7: Add To Stack *****
on mouseUp
set cursor to 4
global finalResult, Place
put "stackBuilder" into Place

--finalResult is the end product of getFinalResult
--It holds the result of the constrained search of the database.
--getFinalResult can be found in the script
--of cd button "query search"

--Place is either "stackBuilder" or "query"
--This makes sure that the info.
--gets put into the correct containers (hidden and visible fields).

set hilite of target to true
ask "Name the result."
if it is empty then
    reset
else
    put it into queryName --if it is not empty
    put queryName & return after card field theStack
    put number of lines in finalResult & return after card field images
    if last character of finalResult = return then
        delete last character of finalResult
    end if
    put finalResult & "." & return after cd field "itemHolder"
    --"itemHolder" is the hidden field that holds the card ids
    --and frame numbers of the items in the Stack.
    reset
end if
end mouseUp
** CARD ** 2, BUTTON #8: Remove *****
on mouseUp

set hilite of target to true
-- Check to see if any buttons are hilited (fields selected)...
if (hilite of card button 1 is false) and ~
(hilite of card button 2 is false) and ~
(hilite of card button 3 is false) and ~
(hilite of card button 4 is false) and ~
(hilite of card button 5 is false) then
    --If nothing has been selected...
    answer "Click on a field to select it, then clear." with "OK"
    set hilite of target to false
    exit mouseUp

--Clear the selected field
--and put "no entry" into the correct hidden field
else
repeat with x = 1 to 5
    if hilite of card button x is true then
        put empty into cd field x
    end if
end repeat
end mouseUp

```

```

        put x into temp
        add 5 to temp
        put empty into cd field temp
        put "no entry" into cd field ("list"&x)
        set hilite of card button x to false
        set hilite of target to false
        exit repeat
      end if
    end repeat
  end if
end mouseUp

** CARD ** 2, BUTTON #9: Block7 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #10: Search *****
on mouseUp
  global Place, theNum
  put "query" into place

  — Place is either "stackBuilder" or "query"
  — This makes sure that the info.
  — gets put into the correct containers (hidden fields).

  — theNum represents both the line in field "theStack"
  — that has been selected and the number of the corresponding item
  — in hidden card field "itemHolder."

  show cd btn "Title"
  show cd btn "Block1"
  show cd btn "Client"
  show cd btn "Block2"
  show cd btn "Medium"
  show cd btn "Block3"
  show cd btn "Subject"
  show cd btn "Block4"
  show cd btn "Box#"
  show cd btn "Block5"
  show cd btn "File"
  show cd btn "Block6"

  — Hide all options other than those necessary for the query
  — Show query fields and buttons

  hide card field theSelection
  hide card button queryCover
  repeat with n = 1 to 5
    show cd field n
    put n into temp
    add 5 to temp
    show cd field temp
    show cd button n
  end repeat
  show cd button "cancel"
  show cd button "query search"

```

```

show cd button "remove"
hide cd field theStack
hide cd field images
repeat with z = 27 to 28
  hide cd button z
end repeat
hide cd btn "Data Cards"
set hilite of target to false
end mouseUp

** CARD ** 2, BUTTON #11: Cancel *****
on mouseUp
set cursor to 4
set hilite of target to true
repeat with x = 1 to 6 --Hilite the correct button
  if ((the short name of cd button x of cd "Main") = "cancel") then
    set hilite of cd button x to true
  else
    set hilite of cd button x of cd "Main" to false
  end if
end repeat
reset
end mouseUp

** CARD ** 2, BUTTON #12: Main *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #14: Quit *****
on mouseUp
reset
domenu "Quit Hypercard"
end mouseUp

** CARD ** 2, BUTTON #15 *****
on mouseUp
send mouseUp to cd field theStack
end mouseUp

** CARD ** 2, BUTTON #16 *****
on mouseUp
send mouseUp to cd field theStack
end mouseUp

** CARD ** 2, BUTTON #17 *****
on mouseUp
send mouseUp to cd field theStack
end mouseUp

** CARD ** 2, BUTTON #18 *****
on mouseUp
send mouseUp to cd field theStack
end mouseUp

** CARD ** 2, BUTTON #19 *****

```

```
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #20 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #21 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #22 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #23 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #24 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #25 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #26 -----
on mouseUp
  send mouseUp to cd field theStack
end mouseUp
```

```
-- CARD -- 2, BUTTON #27: Remove -----
on mouseUp
  global theNum
```

—The user must select a line with 1 click
—or double click in field theStack to select all
—before it can be removed.

—ItemHolder is a hidden field with all of the card ids
—and frame numbers in items - separated by commas.

—theNum represents both the line in field "theStack"
—that has been selected and the number of the corresponding item
—in hidden card field "itemHolder."

```
set hilite of target to true
set cursor to 4
```

```

if theNum is empty then
  answer "Click or double click, select items to remove" with "OK"
  set hilite of target to false
  exit mouseUp
else
  if theNum is "selectAll" then
    put empty into cd field theStack
    put empty into cd field images
    put empty into cd field "itemHolder"
  else
    set lockScreen to true
    delete line theNum of card field theStack
    delete line theNum of card field images
    delete item theNum of card field "itemHolder"
    repeat while (line 1 of cd field itemHolder is empty) and -
      (cd field "itemHolder" is not empty)
      delete line 1 of cd field "itemHolder"
    end repeat
  end if
end if

repeat with x = 15 to 26
  set hilite of cd button x to false
end repeat
repeat with x = 1 to 6
  set hilite of bg button x to false
end repeat

if card field theStack is empty then
  repeat with x = 27 to 28
    hide card button x
  end repeat
  hide cd btn "Data Cards"
  hide cd field theStack
  hide cd field images
  put empty into card field "itemHolder"
end if
set hilite of target to false
put empty into theNum
end mouseUp

```

```

-- CARD -- 2, BUTTON #28: Save -----
--There are 2 ways to save a file: as a unified list or an itemized list
--A unified list has no duplicates
--and is represented with a name given to the union by the user.
--When reading a unified file back into the stack it appears as 1 item.
--An itemized file will read into the stack
--exactly as it was originally - an itemized list.

```

```

on mouseUp
  global theNum, stackEntry, imageTotal, savedItems
  global unionResult, fileName

  if visible of cd button cover is true then
    repeat with x = 1 to 5
      set hilite of bg button x to false
    end repeat
  end if

```

```
hide cd field theSelection
hide cd button cover
end if
```

```
set hilite of target to true
```

```
— theNum represents both the line in field "theStack"
— that has been selected and the number of the corresponding item
— in hidden card field "itemHolder."
```

```
—The PathName xfcn written by Andrew Gilmartin at Brown University.
—PathName returns a full pathname for the file name
—StackEntry is the short version of the file name
```

```
—imageTotal is the number of images being saved.
—In a unified file it is 1 number.
—Otherwise it is an itemized list.
```

```
—savedItems is a list of the card ids and frame numbers being saved.
```

```
—UnionResult is returned
—by the function PlusList found in the script of this stack
```

```
if theNum is empty then
  answer "Click on an item in the Stack to save" with "OK"
  set hilite of target to false
  exit mouseUp
end if
```

```
—The user can save 1 item
—or double click to select all items and save the whole Stack.
```

```
if theNum is "selectAll" then
  answer "Save Stack as unified list (no dupa), or as is..." with ~
  "Union" or "As is" or "cancel"
  if it is "cancel" then
    choose browse tool
    put empty into theNum
    repeat with x = 1 to 26
      set hilite of cd button x to false
    end repeat
    set hilite of target to false
    exit mouseUp
  end if
end if
```

```
else
  if it is "union" then
    set cursor to 4
```

```
    getUnion
    if line 1 of unionResult is empty then
      delete line 1 of unionResult
    end if
    put number of lines in unionResult into imageTotal
    put unionResult into savedItems
```

```
    get newPathName("Save as:",empty)
    if it is empty then
      choose browse tool
```



```
    put empty into theNum
    set hilite of target to false
    send mouseUp to card StackBuilder
    exit mouseUp
  else
    put it into fileName
  end if
  put fileName into stackEntry
  repeat with x = 1 to number of characters in stackEntry
    —to get the short name of the file
    if "." is in stackEntry then
      delete first character of stackEntry
    else
      exit repeat
    end if
  end repeat

  else
    if it is "as is" then
      get newPathName("Save as:",empty)
      if it is empty then
        choose browse tool
        put empty into theNum
        set hilite of target to false
        send mouseUp to card StackBuilder
        exit mouseUp
      else
        put it into fileName
      end if
      put cd field "itemHolder" into savedItems
      put cd field "images" into imageTotal
      put cd field theStack into stackEntry
    end if
  end if
  end if
  saveFile

  else
    if theNum is not "selectAll" then
      put item theNum of cd field "itemHolder" into savedItems
      if line 1 of savedItems is empty then
        delete line 1 of savedItems
      end if
      put the number of lines in savedItems into imageTotal
      put line theNum of cd field theStack into stackEntry
      get newPathName("Save as:",stackEntry)
      if it is empty then
        choose browse tool
        put empty into theNum
        repeat with x = 15 to 26
          set hilite of cd button x to false
        end repeat
        set hilite of target to false
        exit mouseUp
      else
        put it into fileName
        put fileName into stackEntry
      end if
    end if
  end if
end if
```

```

repeat with x = 1 to number of characters in etackEntry
  —to get the short name of the file
  if "." is in stackEntry then
    delete first character of stackEntry
  elsa
    exit repeat
  end if
end repeat
end if
saveFile
end if
end mouseUp

```

```

on saveFile
  —This script uses the same globals as above
  —and actually writes the file to a disk.

```

```

global etackEntry, imageTotal, savedItems, fileName

```

```

if last character of savedItems = return then
  delete last character of savedItems
end if
if last character of imageTotal = return then
  delete last character of imageTotal
end if
if last character of stackEntry = return then
  delete last character of stackEntry
end if

```

```

open file fileName
write stackEntry to file fileName
write "#" to file fileName
write imageTotal to file fileName
write "#" to file fileName
write savedItems to file fileName
close file fileName

```

```

—Reset the card buttons to false
choose browse tool
put empty into theNum
set hilite of target to false
repeat with x = 15 to 26
  set hilite of cd button x to false
end repeat
end saveFile

```

```

on getUnion
  —uses the function plusList found in the script of this stack.
  global unionResult, nextList
  put empty into unionResult
  put cd field "itemHolder" into lists
  put "dummy item" after lists
  put item 1 of lists into unionResult
  delete item 1 of lists
  if line 1 of lists is empty then delete line 1 of lists
  repeat while number of items of lists > 1
    put item 1 of lists into nextList

```

```

    delete item 1 of lists
    put plusList (unionResult, nextList) into unionResult
  end repeat
end getUnion

** CARD ** 2, BUTTON #29: Block1 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #30: Block2 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #31: Block3 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #32: Block4 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #33: Block5 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #34: Block6 *****
on mouseUp

end mouseUp

** CARD ** 2, BUTTON #35: Title *****
on mouseUp
  global cardName, theNum, place
  — cardName directs hyperCard to the correct card holding
  — the contentList and the idList (card ids and frame numbers.)

  — theNum represents both the line in field "theStack"
  — that has been selected and the number of the corresponding item
  — in hidden card field "itemHolder."

  repeat with x = 1 to 5
    set hilite of cd button x to false
  end repeat

  if hilite of target is true then
    — Provides a way out of this procedure

```

```

hide cd field theSelection
hide cd button cover
set hilite of target to false
exit mouseUp
else
put (the short name of me) into cardName
set cursor to 4
repeat with x = 1 to 6 — hilite the correct button
if the short name of bg button x is (the short name of me) then
set hilite of bg button x to true
else
set hilite of bg button x to false
end if
end repeat

get bg field contentList of card cardName
put it into card field "theSelection"
set scroll of card field "theSelection" to 0
show card field "theSelection"
end if
hide cd btn "Query Search"
hide cd btn "Remove"
end mouseUp

** CARD ** 2, BUTTON #36: Client *****
on mouseUp
global cardName, theNum, place
— cardName directs hyperCard to the correct card holding
— the contentList and the idList (card ids and frame numbers.)

— theNum represents both the line in field "theStack"
— that has been selected and the number of the corresponding item
— in hidden card field "itemHolder."

repeat with x = 1 to 5
set hilite of cd button x to false
end repeat

if hilite of target is true then
— Provides a way out of this procedure
hide cd field theSelection
hide cd button cover
set hilite of target to false
exit mouseUp
else
put (the short name of me) into cardName
set cursor to 4
repeat with x = 1 to 6 — hilite the correct button
if the short name of bg button x is (the short name of me) then
set hilite of bg button x to true
else
set hilite of bg button x to false
end if
end repeat

get bg field contentList of card cardName
put it into card field "theSelection"

```

```

set ecroll of card field "theSelection" to 0
show card field "theSelection"
end if
hide cd btn "Query Search"
hide cd btn "Remove"
end mouseUp

```

```

-- CARD -- 2, BUTTON #37: Medium -----

```

```

on mouseUp
global cardName, theNum, place
-- cardName directs hyperCard to the correct card holding
-- the contentList and the idList (card ids and frame numbers.)

-- theNum represents both the line in field "theStack"
-- that has been selected and the number of the corresponding item
-- in hidden card field "itemHolder."

repeat with x = 1 to 5
set hilite of cd button x to false
end repeat

```

```

if hilite of target is true then
-- Provides a way out of this procedure
hide cd field theSelection
hide cd button cover
set hilite of target to false
exit mouseUp
else
put (the short name of me) into cardName
set cursor to 4
repeat with x = 1 to 6 -- hilite the correct button
if the short name of bg button x is (the short name of me) then
set hilite of bg button x to true
else
set hilite of bg button x to false
end if
end repeat

```

```

get bg field contentList of card cardName
put it into card field "theSelection"
set ecroll of card field "theSelection" to 0
show card field "theSelection"
end if
hide cd btn "Query Search"
hide cd btn "Remove"
end mouseUp

```

```

-- CARD -- 2, BUTTON #38: Subject -----

```

```

on mouseUp
global cardName, theNum, place
-- cardName directs hyperCard to the correct card holding
-- the contentList and the idList (card ids and frame numbers.)

-- theNum represents both the line in field "theStack"
-- that has been selected and the number of the corresponding item
-- in hidden card field "itemHolder."

```

```
repeat with x = 1 to 5
  set hilite of cd button x to false
end repeat
```

```
if hilite of target is true then
  — Provides a way out of this procedure
  hide cd field theSelection
  hide cd button cover
  set hilite of target to false
  exit mouseUp
else
  put (the short name of me) into cardName
  set cursor to 4
  repeat with x = 1 to 6 — hilite the correct button
    if the short name of bg button x is (the short name of me) then
      set hilite of bg button x to true
    else
      set hilite of bg button x to false
    end if
  end repeat
```

```
get bg field contentList of card cardName
put it into card field "theSelection"
set scroll of card field "theSelection" to 0
show card field "theSelection"
end if
hide cd btn "Query Search"
hide cd btn "Remove"
end mouseUp
```

```
-- CARD ~ 2, BUTTON #39: Box# ~~~~~
```

```
on mouseUp
  global cardName, theNum, place
  — cardName directs hyperCard to the correct card holding
  — the contentList and the idList (card ids and frame numbers.)

  — theNum represents both the line in field "theStack"
  — that has been selected and the number of the corresponding item
  — in hidden card field "itemHolder."
```

```
repeat with x = 1 to 5
  set hilite of cd button x to false
end repeat
```

```
if hilite of target is true then
  — Provides a way out of this procedure
  hide cd field theSelection
  hide cd button cover
  set hilite of target to false
  exit mouseUp
else
  put "Location" into cardName
  set cursor to 4
```

```

repeat with x = 1 to 6 — hilite the correct button
  if the short name of bg button x is (the short name of me) then
    set hilite of bg button x to true
  else
    set hilite of bg button x to false
  end if
end repeat

```

```

get bg field contentList of card cardName
put it into card field "theSelection"
set scroll of card field "theSelection" to 0
show card field "theSelection"
end if
hide cd btn "Query Search"
hide cd btn "Remove"
end mouseUp

```

```

-- CARD -- 2, BUTTON #40: File *****
on mouseUp

```

—PathName is an xfcn written by Andrew Gilmartin at Brown University
 —It brings up the traditional Apple window for file saving.

```

get PathName("TEXT")
if it is empty then —if operation is cancelled....
  set hilite of target to false
  choose browse tool
  exit mouseUp
else
  —See script of card button "seve" to find out how files are saved.
  put it into fileName
  open file fileName
  read from file fileName until "#"
  delete last char of it
  put it into nameHolder
  read from file fileName until "#"
  delete last char of it
  put it into imageHolder
  read from file fileName until empty
  delete last char of it
  put it into savedItems
  if last line of savedItems is empty then
    delete last line of savedItems
  end if

```

```

put fileName into FieldEntry
—FileName includes the pathName
—This script gets the short name of the file
repeat with x = 1 to number of characters in FieldEntry
  if "." is in FieldEntry then
    delete first character of FieldEntry
  else
    exit repeat
  end if
end repeat

```

```

put FieldEntry after cd fld theStack —Name of the file
put imageHolder after cd fld Images —total number of images
put savedItems & ", " & return after cd fld itemholder

```



```

show cd field theStack
show cd field images
repeat with x = 27 to 28
  show card button x
end repeat
show cd btn "Data Cards"

close file fileName
set hilite of target to false
choose browse tool
end if
end mouseUp

** CARD ** 2, BUTTON #42 *****
on mouseUp
if not visible of cd fld 13 then
  show cd fld 13
  show cd fld 14
  show cd fld 15
  show cd fld 16
  show cd fld 17
  show cd fld 18
else
  hide cd fld 13
  hide cd fld 14
  hide cd fld 15
  hide cd fld 16
  hide cd fld 17
  hide cd fld 18
end if
end mouseUp

** CARD ** 2, BUTTON #44: Data Cards *****
on mouseUp
global theNum, theList
push this cd

— theNum represents both the line in field "theStack"
— that has been selected and the number of the corresponding item
— in hidden card field "itemHolder."

—theList is card ids of the selected item or items from theStack
—theList is used to initialize and place the scrollBox (on data card)
—and is accessed on the goToCard script in the data card stack.

set hilite of target to true
set cursor to 4

if theNum is empty then
  answer "Click or double click to select items first" with "OK"
  set hilite of target to false
  exit mouseUp
else

if theNum = "selectAll" then
  —put the whole field into theList
  put the number of items in cd fld itemHolder into temp

```

```

put empty into theList
repeat with y = 1 to temp
  put item y of cd field "itemHolder" after theList
end repeat
else
  —put just the selected item into theList
  put item theNum of cd field "itemHolder" into theList
end if
end if
if line 1 of theList = empty then delete line 1 of theList

set hilite of target to false
repeat with x = 15 to 26
  set hilite of cd button x to false
end repeat
—The first word of each line of theList is a card id number.
go card id (first word of line 1 of theList) of stack "Data Cards"
put empty into theNum
end mouseUp

** CARD #3: Intro *****
on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "Introduction", 95,32
end idle
** CARD #4 *****
on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "Introduction", 95,32
end idle
** CARD ** 4, BUTTON #1: Introduction *****
on mouseUp

end mouseUp

** CARD #5 *****
on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "Introduction", 95,32
end idle
** CARD ** 5, BUTTON #1: Introduction *****
on mouseUp

end mouseUp

** CARD #6 *****
on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "Introduction", 95,32
end idle
** CARD ** 6, BUTTON #1: Introduction *****
on mouseUp

```

end mouseUp

"" CARD "" 6, BUTTON #2: Return ""
on mouseUp
go to cd "Main"
end mouseUp

"" CARD #7: Bio ""
on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 95,8
showpict "Biographical Information", 94,31
showpict "Beall.1", 227,65
end idle

"" CARD "" 7, BUTTON #1: Biography ""
on mouseUp

end mouseUp

"" CARD #8 ""
on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 95,8
showpict "continued...", 93,31
showpict "Chicago.3", 305,65
end idle

"" CARD "" 8, BUTTON #1: Biography ""
on mouseUp

end mouseUp

"" CARD #9 ""
on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 95,8
showpict "continued...", 93,31
showpict "Chrysler", 105,62
end idle

"" CARD "" 9, BUTTON #1: Biography ""
on mouseUp

end mouseUp

"" CARD #10 ""
on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 95,8
showpict "continued...", 93,31
showpict "Dunbarton", 204,66

end idle

~~ CARD ** 10, BUTTON #1: Biography ~~~~~
on mouseUp

end mouseUp

~~ CARD #11 ~~~~~

on idle
 showpict "GDA LOGO.2", 15,9
 showpict "Lester Beall", 95,8
 showpict "continued...", 93,31
end idle

~~ CARD ** 11, BUTTON #1: Return ~~~~~

on mouseUp
 go to cd "Main"
end mouseUp

~~ CARD ** 11, BUTTON #3: Biography ~~~~~

on mouseUp

end mouseUp

SCRIPTS FOR STACK: Time Line

~~ STACK SCRIPT ~~~~~

on idle
 global counter
 put the mouseLoc into mouseCheck
 wait 6 ticks
 if the mouseLoc <> mouseCheck then
 put 0 into counter
 else
 put counter + 1 into counter
 end if
 if counter = 3000 then
 put 0 into counter
 go to cd 1 of stack "Beall Archive"
 end if
end idle

on closestack
 get the size of this stack
 put it into StackSize
 get the freesize of this stack
 put it into FreeSpace
 if FreeSpace > (StackSize *.15) then
 domenu "Compact Stack"
 end if
end closestack

on domenu command

```
if command = "Quit Hypercard" then
  go cd Main of stack "Beall Archive"
  exit domenu
end if
pass domenu
end domenu
```

—Graphic Design Archive Stackware™
—Copyright 1989 Rochester Institute of Technology
—Portions copyright 1988/1987 Apple Computer, Inc.
—Unless otherwise noted all scripts written by Cathleen Britt at RIT.

```
on openStack
  hide menuber
  hide mag
  global initialized
```

—Initialized is a boolean expression (true or false)
—It insures that the user always opens the GDA Archive
—through the "GDA Start" stack
—and that the necessary globals (xcmds) are declared.
—If it is true that means that the globals in stack "GDA Start"
—have been declared and that the video drivers are working.
—If it is not true then go to stack "GDA Start" and openStack there.

—Index is the number of the card that is open
—(number of the line in theList)
—The index number is shown on the card in the scroll box

—IndexHolder keeps track of the number of the video image
—that is on the screen when the user has stopped scanning.
—It allows the user to scan images, stop on a particular one
—and go to the correct data card for that image.

—Source is either "scan" or "reference".
—This script checks to see what source is
—to determine whether or not to initialize the scroll box
—and where to place it.

```
if initialized is not true then —to declare globals...
  go to stack "Beall Archive"
end if
end openStack
```

```
** BKGND:#1, BUTTON #1: Block 1 *****
on mouseUp

end mouseUp
```

```
** BKGND:#1, BUTTON #2: Block2 *****
on mouseUp

end mouseUp
```

```
** BKGND:#1, BUTTON #3: Block3 *****
on mouseUp
```

```

end mouseUp

** BKGND!#1, BUTTON #4: Block4 *****
on mouseUp

end mouseUp

** CARD #1 *****
on DecadeFields
global Topic
global TlDate
global PrevPos
put cd fld "slider" into TlDate
put Topic&&TlDate into TheField
show cd fld TheField
if Topic = "Quotes" then
  ShowQuotes
end if
if (TlDate <> PrevPos) and (PrevPos <> empty) then
  put Topic&&PrevPos into TheField
  hide cd fld TheField
end if
end DecadeFields

on opencard
global Topic
repeat with i = 1 to 4
  if the hilite of cd btn i is true then
    put the short name of cd btn i into Topic
  end if
end repeat
end opencard

on ShowQuotes
global TlDate
if TlDate = 1940 then
  show cd btn "Page 1 Quotes"
  show cd btn "Page 2 Quotes"
  hide cd btn "Page 4 Quotes"
  hide cd btn "Page 3 Quotes"
  send mouseUp to cd btn "Page 1 Quotes"
else if TlDate = 1950 or TlDate = 1960 then
  show cd btn "Page 1 Quotes"
  show cd btn "Page 2 Quotes"
  show cd btn "Page 3 Quotes"
  show cd btn "Page 4 Quotes"
  send mouseUp to cd btn "Page 1 Quotes"
else
  QuoteButtons
end if
end showQuotes

on QuoteButtons
lock screen
hide cd btn "Page 1 Quotes"

```

```

hide cd btn "Page 2 Quotes"
hide cd btn "Page 3 Quotes"
hide cd btn "Page 4 Quotes"
unlock screen
end QuoteButtons

on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 95,8
showpict "Time Line", 93,31
if visible of cd fld "Top Bar" then
  showpict "Navigator.ICON.2", 18,252
else
  showpict "Navigator.ICON", 18,252
end if

—put (round((the FreeSize of this stack) /1024)&" K") into~
— cd fld "Free"
end idle

on hideButtons
lock screen
hide cd btn "Cover"
hide cd fld "Top Bar"
hide cd btn "Put Away" —Put away box
hide cd btn "Intro.Nav"
hide cd btn "Bio.Nav"
hide cd btn "Query.Nav"
hide cd btn "Time Line.Nav"
hide cd btn "Data Cards.Nav"
hide cd btn "Main.Nav"
hide cd btn "Rectangle"
unlock screen
open card
end hideButtons

"" CARD #1, FIELD #1: Biographic 1900 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #2: Biographic 1910 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #3: Biographic 1920 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #4: Biographic 1930 *****
on mouseUp
set the scroll of me to 0
hide me

```

```

end mouseUp

** CARD #1, FIELD #5: Biographic 1940 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #6: Biographic 1950 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #7: Biographic 1960 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp
** CARD #1, FIELD #8: Biographic 1970 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #9 *****
on mouseDown
  —
  — Compute the Minimum and Maximum X values from the Scrollbar rect.
  — Grab and preserve starting point, then enter the repeat loop as
  — long as the mouse is still down.
  —
  global PrevPos
  put left of cd btn "Scrollbar" + 19 into MinX
  put right of cd btn "Scrollbar" - 19 into MaxX
  put cd fld "Slider" into PrevPos
  put loc of me into newLoc
  repeat while the mouse is down
    —
    — Test to see if the mouse is less than the maximum X.
    — If so, then proceed, otherwise set loc of elevator to
    — maximum setting.
    —
    if the mouseH < (MaxX+1) then
      —
      — Test to see if the mouse is less than the minimum X.
      — If so, then set loc of elevator to minimum setting.
      — If not, then set loc of elevator to mouse's X position.
      —
      if the mouseH < MinX then
        put MinX into item 1 of newLoc
        set loc of me to newLoc
      else
        put the mouseH into item 1 of newLoc
        set loc of me to newLoc
      end if
    —
    — Set the value in the display field

```

```

--
setit
else
--
-- Set elevator to maximum value
--
put MaxX into item 1 of newLoc
set loc of me to newLoc
--
-- Set the value in the display field
--
setit
end if
end repeat
DecadeFields
end mouseDown

on setit
--
-- Compute the Minimum X and Difference X values from the Scrollbar.
-- Grab the maximum numeric value from the field.
--
put left of cd btn "Scrollbar" into MinX
put right of cd btn "Scrollbar" - left of cd btn "Scrollbar" - 10--
into DifX
put 7 into MaxVal
--
-- Compute the correct setting for the displayed value
--
put round ((the mouseH - MinX)/DifX*MaxVal) into holder
if holder ≤ 0 then put 0 into holder
if holder ≥ MaxVal then put MaxVal into holder
put 19&holder&0 into line 1 of cd fld "slider"
end setit
** CARD #1, FIELD #10: Historic 1900 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

** CARD #1, FIELD #11: Historic 1910 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

** CARD #1, FIELD #12: Historic 1920 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

** CARD #1, FIELD #13: Historic 1930 *****
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

```

```
** CARD #1, FIELD #14: Historic 1940 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #15: Historic 1950 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #16: Historic 1960 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #17: Historic 1970 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #18: Client Info 1900 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #19: Client Info 1910 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #20: Client Info 1920 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #21: Client Info 1930 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #22: Client Info 1940 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

** CARD #1, FIELD #23: Client Info 1950 *****
on mouseUp
  set the scroll of me to 0
  hide me
```

set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #34: Quotes 1940.1
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #35: Quotes 1940.2
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #36: Quotes 1950.1
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #37: Quotes 1950.2
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #38: Quotes 1950.3
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #39: Quotes 1950.4
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #40: Quotes 1960.1
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #41: Quotes 1960.2
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

"" CARD #1, FIELD #42: Quotes 1960.3
on mouseUp
set the scroll of me to 0
hide me
end mouseUp

```

** CARD #1, FIELD #43: Quotes 1960.4 *****
on mouseUp
  set the scroll of me to 0
  hide me
end mouseUp

```

```

** CARD #1, FIELD #44: Top Bar *****
on mouseStillDown
  global diffx,diffy
  put item 1 of the mouseLoc into x
  put item 2 of the mouseLoc into y
  if y > 60 and x > 100 then
    show me at x-diffx,y-diffy
    show cd btn "Rectangle" at ((x-diffx)+1),(y-diffy)+128)
  end if
end mouseStillDown

```

```

on mousedown
  global diffx,diffy
  put item 1 of the loc of me into CenterPtX
  put item 2 of the loc of me into CenterPtY
  put item 1 of the mouseLoc into x
  put item 2 of the mouseLoc into y
  put (x-CenterPtX) into diffx
  put (y-CenterPtY) into diffy
end mousedown

```

```

on mouseUp
  put item 1 of the loc of me into x
  put item 2 of the loc of me into y
  lock screen
  show cd btn "Put Away" at x-102,y+5 —Put away box
  show cd btn "Intro.Nav" at x+0,y+38
  show cd btn "Bio.Nav" at x+92,y+106
  show cd btn "Query.Nav" at x+60,y+210
  show cd btn "Time Line.Nav" at x-59,y+210
  show cd btn "Data Cards.Nav" at x-92, y+106
  show cd btn "Main.Nav" at x+0,y+135
  unlock screen
  showpict "Navigator", x-113,y-6
end mouseUp

```

```

** CARD ** 1, BUTTON #1: Client Info *****
on mouseUp
  global Topic
  put Topic into PrevDec
  set the hilite of cd btn "Biographic" to false
  set the hilite of cd btn "Historic" to false
  set the hilite of cd btn "Quotes" to false
  set the hilite of me to true
  lock screen
  QuoteButtons
  put the short name of me into Topic
  DecadeFields
  put PrevDec&&cd fld "slider" into ShowingField
  hide cd fld ShowingField
end mouseUp

```

```

** CARD ** 1, BUTTON #2: Historic *****

```



```

put right of cd btn "Scrollbar" - left of cd btn "Scrollbar" - 20-
into DifX
put 0 into MinPos
put 7 into MaxPos
put cd fld "Slider" into PrevPos
repeat while the mouse = down
—
— Determine next value for display field, then set the loc of
— the elevator to the correct X position for that value.
—
put character 3 of cd fld "slider" + 1 into NextPos
if NextPos ≥ MaxPos
then
set loc of cd fld "slider" to MaxX,-
item 2 of loc of cd fld "slider"
put 19&MaxPos&0 into cd fld "slider"
else
set loc of cd fld "slider" -
to round ((NextPos/MaxPos) * DifX + MinX),-
item 2 of loc of cd fld "slider"
put cd fld "Slider" + 10 into cd fld "slider"
end if
end repeat
DecadeFields
end mouseDown
** CARD ** 1, BUTTON #8: LeftArrow *****
on mouseDown
—
— Compute the Minimum, Maximum and Difference X values using the
— Scrollbar's rect, then enter the repeat loop as
— long as the mouse is still down.
—
global PrevPos
put left of cd btn "Scrollbar" + 19 into MinX
put right of cd btn "Scrollbar" - 19 into MaxX
put right of cd btn "Scrollbar" - left of cd btn "Scrollbar" - 21-
into DifX
put 0 into MinPos
put 7 into MaxPos
put cd fld "Slider" into PrevPos
repeat while the mouse = down
—
— Determine next value for display field, then set the loc of
— the elevator to the correct X position for that value.
—
put character 3 of cd fld "slider" - 1 into NextPos
if NextPos ≤ MinPos then
set the loc of cd fld "slider" to MinX,-
item 2 of loc of cd fld "slider"
put 19&MinPos&0 into cd fld "slider"
else
set loc of cd fld "slider" -
to round ((NextPos/MaxPos) * DifX + MinX),-
item 2 of loc of cd fld "slider"
put cd fld "Slider" - 10 into cd fld "slider"
end if
end repeat
DecadeFields

```

```

end mouseDown
** CARD ** 1, BUTTON #9: Page 1 Quotes *****
on mouseUp
  global Decade
  global TDate
  set the hilite of cd btn "Page 2 Quotes" to false
  set the hilite of cd btn "Page 3 Quotes" to false
  set the hilite of cd btn "Page 4 Quotes" to false
  set the hilite of me to true
  put "Quotes " & TDate & ".1" into Hidden
  put "Quotes " & TDate into TheField
  — put the hidden field (Historical 19xx.x) into the open one
  lock screen
  put cd fld Hidden into cd fld TheField
  set the scroll of cd fld TheField to 0
  unlock screen
end mouseUp

```

```

** CARD ** 1, BUTTON #10: Page 2 Quotes *****
on mouseUp
  global Decade
  global TDate
  set the hilite of cd btn "Page 1 Quotes" to false
  set the hilite of cd btn "Page 3 Quotes" to false
  set the hilite of cd btn "Page 4 Quotes" to false
  set the hilite of me to true
  put "Quotes " & TDate & ".2" into Hidden
  put "Quotes " & TDate into TheField
  — put the hidden field (Historical 19xx.x) into the open one
  lock screen
  put cd fld Hidden into cd fld TheField
  set the scroll of cd fld TheField to 0
  unlock screen
end mouseUp

```

```

** CARD ** 1, BUTTON #11: Page 3 Quotes *****
on mouseUp
  global Decade
  global TDate
  set the hilite of cd btn "Page 2 Quotes" to false
  set the hilite of cd btn "Page 1 Quotes" to false
  set the hilite of cd btn "Page 4 Quotes" to false
  set the hilite of me to true
  put "Quotes " & TDate & ".3" into Hidden
  put "Quotes " & TDate into TheField
  — put the hidden field (Historical 19xx.x) into the open one
  lock screen
  put cd fld Hidden into cd fld TheField
  set the scroll of cd fld TheField to 0
  unlock screen
end mouseUp

```

```

** CARD ** 1, BUTTON #12: Page 4 Quotes *****
on mouseUp
  global Decade
  global TDate
  set the hilite of cd btn "Page 2 Quotes" to false
  set the hilite of cd btn "Page 3 Quotes" to false

```



```

set the hilite of cd btn "Page 1 Quotes" to false
set the hilite of me to true
put "Quotes " & TDate & ".4" into Hidden
put "Quotes " & TDate into TheField
— put the hidden field (Historical 19xx.x) into the open one
lock screen
put cd fld Hidden into cd fld TheField
set the scroll of cd fld TheField to 0
unlock screen
end mouseUp

** CARD ** 1, BUTTON #13: Lock or Unlock Field *****
on mouseDown
—
— Preserve the original state of the msg, so we can restore it
— when done, set up a wait period, then toggle the cursor back
— and forth to the two options. When user selects option, then
— put an explanatory note into msg box.
—
— put the visible of msg into morigVis
put 20 into lag
repeat until the mouse = "up"
  set cursor to "Lock.2"
  wait lag
  if the mouse = "up" then
    —put "Click in the field to LOCK. (holding longer toggles to UNLOCK)."
    put "true" into action
    exit repeat
  end if
  set cursor to "Lock.1"
  wait lag
  if the mouse = "up" then
    —put "Click in the field to UNLOCK"~
    —&& "(holding longer toggles to LOCK)."
    put "false" into action
    exit repeat
  end if
end repeat
wait until the mouse = "down"
put the mouseLoc into testHere
—
— Step through all of the card fields and test to see if cursor
— was clicked within its rect and that the field is visible.
— If so, then change the lockText, restore msg, quit routine.
—
repeat with n = 1 to number of cd flds
  if testHere is within rect of cd fld n and~
  the visible of cd fld n = "true" then
    set the locktext of cd fld n to action
    — set the visible of msg to morigVis
    exit mouseDown
  end if
end repeat
—
— The following routine is commented out for use within this stack.
— Just remove the comment lines to make this button work on bkgnd
— fields as well as card fields.
—

```

```

— repeat with n = 1 to the number of bg flds
— if testHere is within rect of bg fld n and-
— the visible of bg fld n = "true" then
— set the locktext of bg fld n to action
— set the visible of msg to morigVis
— exit mouseDown
— end if
— end repeat
— set the visible of msg to morigVis
end mouseDown

```

```

** CARD ** 1, BUTTON #14: Cover *****
on mouseUp
play boing
end mouseUp

```

```

** CARD ** 1, BUTTON #15: Compass *****
on mouseUp
if not visible of cd fld "Top Bar" then
show cd btn "Cover"
show cd fld "Top Bar" at 299,67
put item 1 of the loc of cd fld "Top Bar" into x
put item 2 of the loc of cd fld "Top Bar" into y
lock screen
show cd btn "Rectangle" at 300,195
show cd btn "Put Away" at x-102,y+5 —Put away box
show cd btn "Intro.Nav" at x+0,y+38
show cd btn "Bio.Nav" at x+92,y+106
show cd btn "Query.Nav" at x+60,y+210
show cd btn "Time Line.Nav" at x-59,y+210
show cd btn "Data Cards.Nav" at x-92, y+106
show cd btn "Main.Nav" at x+0,y+135
unlock screen
showpict "Navigator", x-113,y-6
else if visible of cd fld "Top Bar" then
hideButtons
end if
end mouseUp

```

```

** CARD ** 1, BUTTON #16: Put Away *****
on mouseup
lock screen
hide me
hideButtons
end mouseup

```

```

** CARD ** 1, BUTTON #18: Intro.Nav *****
on mouseUp
hideButtons
go to cd Intro of stack "Beall Archive"
end mouseUp

```

```

** CARD ** 1, BUTTON #19: Bio.Nav *****
on mouseUp
lock screen
hideButtons
unlock screen
go to cd Bio of stack "Beall Archive"
end mouseUp

```

```

** CARD ** 1, BUTTON #20: Query.Nav *****
on mouseUp
hideButtons
go to stack "Quotes"
end mouseUp

```

```

** CARD ** 1, BUTTON #21: Time Line.Nav *****
on mouseUp
push this cd
hideButtons
go to stack "Time Line"
end mouseUp

```

```

** CARD ** 1, BUTTON #22: Data Cards.Nav *****
on mouseUp
push this cd
hideButtons
go to stack "Data Cards"
end mouseUp

```

```

** CARD ** 1, BUTTON #23: Main.Nav *****
on mouseUp
hideButtons
go to cd Main of stack "Beall Archive"
end mouseUp

```

```

** CARD ** 1, BUTTON #24: Return *****
on mouseUp
pop card
end mouseUp

```

```

** CARD ** 1, BUTTON #25: Biographic *****
on mouseUp
global Topic
put Topic into PrevDec
set the hilite of cd btn "Client Info" to false
set the hilite of cd btn "Historic" to false
set the hilite of cd btn "Quotes" to false
set the hilite of me to true
QuoteButtons
put the short name of me into Topic
DecadeFields
put PrevDec&&cd fld "slider" into ShowingField
hide cd fld ShowingField
end mouseUp

```

SCRIPTS FOR STACK: Quotes

=====

```

** STACK SCRIPT *****
on openStack
hide menubar
hide msg
global initialized

```

- Initialized is a boolean expression (true or false)
- It insures that the user always opens the GDA Archive
- through the "GDA Start" stack
- and that the necessary globals (xcmds) are declared.
- If it is true that means that the globals in stack "GDA Start"
- have been declared and that the video drivers are working.
- If it is not true then go to stack "GDA Start" and openStack there.

```

if initialized is not true then —to declare globals...
  go to stack "Beall Archive"
else
  put 1 into bg fld "Slider"
  show bg fld "Slider" at 137,317
end if

```

```
end openStack
```

```

on idle
  global counter
  put the mouseLoc into mouseCheck
  wait 6 ticks
  if the mouseLoc <> mouseCheck then
    put 0 into counter
  else
    put counter + 1 into counter
  end if
  if counter = 3000 then
    put 0 into counter
    go to cd 1 of stack "Beall Archive"
  end if
end idle

```

```

on domenu command
  if command = "Quit Hypercard" then
    go to cd "Main" of stack "Beall Archive"
    exit domenu
  end if
  pass domenu
end domenu

```

```

-- BACKGROUND #1: Research -----
on opencard
  —sets the scrollbar
  put the number of this cd into record
  if bg fld slider <> record then
    put record into bg fld "slider"
  end if
end opencard

```

```

on idle
  showpict "GDA LOGO.2", 15,9
  showpict "Lester Beall", 95,8
  showpict "Quotes", 95,32
  pass idle
end idle
-- BKGND #1, FIELD #7 -----
on mouseDown
  — Taken from "101 Scripts and Buttons

```

```

— Compute the Minimum and Maximum X values from the Scrollbar rect.
— Grab and preserve starting point, then enter the repeat loop as
— long as the mouse is still down.
—
put left of bg btn "Scrollbar" + 24 into MinX
put right of bg btn "Scrollbar" - 25 into MaxX
put loc of me into newLoc
repeat while the mouse is down
—
— Test to see if the mouse is less than the maximum X.
— If so, then proceed, otherwise set loc of elevator to
— maximum setting.
—
if the mouseH < (MaxX+1) then
—
— Test to see if the mouse is less than the minimum X.
— If so, then set loc of elevator to minimum setting.
— If not, then set loc of elevator to mouse's X position.
—
if the mouseH < MinX then
put MinX into item 1 of newLoc
set loc of me to newLoc
else
put the mouseH into item 1 of newLoc
set loc of me to newLoc
end if
—
— Set the value in the display field
—
setit
else
—
— Set elevator to maximum value
—
put MaxX into item 1 of newLoc
set loc of me to newLoc
—
— Set the value in the display field
—
setit
end if
end repeat
lock screen
go to cd (bg fld "slider")
—sets the scrollbar to the correct card no.
— put (the number of this cd - 1) into bg fld "slider"
unlock screen with visual effect wipe left
end mouseDown

on setit
—
— Compute the Minimum X and Difference X values from the Scrollbar.
— Grab the maximum numeric value from the field.
—
put left of bg btn "Scrollbar" into MinX
put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 10~
into DifX
put number of cards -1 into MaxVal

```

```

—
— Compute the correct setting for the displayed value
—
put round ((the mouseH - MinX)/DiX*MaxVal) into holder
if holder ≤ 1 then put 1 into holder
if holder ≥ MaxVal then put MaxVal into holder
put holder into line 1 of bg fld "slider"
end setit
** BKGND!#1, BUTTON #1: Main *****
on mouseUp
go to cd "Main" of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #2: Introduction *****
on mouseUp
go to cd "Intro" of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #3: Biography *****
on mouseUp
go to cd "Bio" of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #4: Time Line *****
on mouseUp
push this cd
go to stack "Time Line"
end mouseUp

** BKGND!#1, BUTTON #5: ScrollBar *****
on mouseDown
— Taken from "101 Scripts and Buttons
put left of bg btn "Scrollbar" + 24 into MinX
put right of bg btn "Scrollbar" - 25 into MaxX
put item 1 of the clickLoc & ","~
& item 2 of loc of bg fld "slider" into newLoc
if item 1 of the clickLoc < MinX then
put MinX into item 1 of newLoc
else if item 1 of the clickLoc > MaxX then
put MaxX into item 1 of newLoc
end if
set loc of bg fld "slider" to newLoc
send setit to bg fld "slider"
lock screen
go to cd (bg fld "slider" + 1)
—sets the scrollbar to the correct card no.
put (the number of this cd - 1) into bg fld "slider"
unlock screen with visual effect wipe left
end mouseDown
** BKGND!#1, BUTTON #7: LeftArrow *****
on mouseDown
— Taken from "101 Scripts and Buttons"
— Compute the Minimum, Maximum and Difference X values using the
— Scrollbar's rect, then enter the repeat loop as

```

```

— long as the mouse is still down.
—
put left of bg btn "Scrollbar" + 24 into MinX
put right of bg btn "Scrollbar" - 25 into MaxX
put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 50↵
into DifX
put one into MinPos
put number of cards -1 into MaxPos
repeat while the mouse = down
—
— Determine next value for display field, then set the loc of
— the elevator to the correct X position for that value.
—
put bg fld "slider" - 1 into NextPos
if NextPos ≤ MinPos then
  set the loc of bg fld "slider" to MinX,↵
  item 2 of loc of bg fld "slider"
  put "1" into bg fld "slider"
else
  put NextPos into line 1 of bg fld "slider"
  set loc of bg fld "slider" ↵
  to round ((NextPos/MaxPos) * DifX + MinX),↵
  item 2 of loc of bg fld "slider"
end if
end repeat
lock screen
go to cd (bg fld "slider" + 1)
—sets the scrollbar to the correct card no.
put (the number of this cd - 1) into bg fld "slider"
unlock screen with visual effect wipe right
end mouseDown
** BKGND!#1, BUTTON #9: RightArrow *****
on mouseDown
— Taken from "101 Scripts and Buttons
— Compute the Minimum, Maximum and Difference X values using the
— Scrollbar's rect, then enter the repeat loop as
— long as the mouse is still down.
—
put left of bg btn "Scrollbar" + 24 into MinX
put right of bg btn "Scrollbar" - 25 into MaxX
put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 50↵
into DifX
put one into MinPos
put number of cards -1 into MaxPos
repeat while the mouse = down
—
— Determine next value for display field, then set the loc of
— the elevator to the correct X position for that value.
—
put bg fld "slider" + 1 into NextPos
if NextPos ≥ MaxPos
then
  put MaxPos into bg fld "slider"
  set loc of bg fld "slider" to MaxX,↵
  item 2 of loc of bg fld "slider"
else
  put NextPos into bg fld "slider"
  set loc of bg fld "slider" ↵

```

```

    to round ((NextPos/MaxPos) * DifX + MinX),-
    item 2 of loc of bg fld "slider"
end if
end repeat
lock screen
go to cd (bg fld "slider" + 1)
—sets the scrollbar to the correct card no.
put (the number of this cd - 1) into bg fld "slider"
unlock screen with visual effect wipe left
end mouseDown
** BKGND!#1, BUTTON #10: Quotes *****
on mouseUp

end mouseUp

```

SCRIPTS FOR STACK: Data Cards

```

=====
** STACK SCRIPT *****
—Graphic Design Archive Stackware™
—Copyright 1989 Rochester Institute of Technology
—Portions copyright 1988/1987 Apple Computer, Inc.
—Unless otherwise noted all scripts written by Cathleen Britt at RIT.

```

```

on idle
global counter
global index
put index into bg fld "scrollbox"
put the mouseLoc into mouseCheck
wait 6 ticks
if the mouseLoc <> mouseCheck then
    put 0 into counter
else
    put counter + 1 into counter
end if
if counter = 3000 then
    put 0 into counter
    go to cd 1 of stack "Beall Archive"
end if
end idle

```

```

on openStack
hide menubar
hide msg
global initialized, index, indexHolder, source

```

```

—Initialized is a boolean expression (true or false)
—It insures that the user always opens the GDA Archive
—through the "GDA Start" stack
—and that the necessary globals (xcmds) are declared.
—If it is true that means that the globals in stack "GDA Start"
—have been declared and that the video drivers are working.
—If it is not true then go to stack "GDA Start" and openStack there.

```

```

—Index is the number of the card that is open
—(number of the line in theList)
—The index number is shown on the card in the scroll box

```

—IndexHolder keeps track of the number of the video image
—that is on the screen when the user has stopped scanning.
—It allows the user to scan images, stop on a particular one
—and go to the correct data card for that image.

—Source is either "scan" or "reference".
—This script checks to see what source is
—to determine whether or not to initialize the scroll box
—and where to place it.
if initialized is not true then —to declare globals...
 go to stack "Beall Archive"
else
 —if returning to card stack from reference stack...
 if source is "reference" then —return to the correct card...
 put indexHolder into index —show the correct index number...
 put empty into source
 else
 —if source is scan then the scroll bar is initialized
 —with a new list and indexHolder determines
 —the placement of the scroll box.
 if source is "scan" then
 initializeScrollBar
 put indexHolder into index —get correct index number
 goToCard —go to correct card
 put empty into source
 else
 —go to first card id on theList
 hide menuBar
 initializeScrollBar
 goToCard
 end if
 end if
end if
put 1 into bg fld "Slider"
show bg fld "Slider" at 45,321
end openStack

on openCard
 global playerType
 —PlayerType -the type of videoDisc player that is being used.

PlaceScrollBar
—If no videoDisc player is being used...
if playerType = "none" then —disables searchVideo command
 exit openCard
else
 —each card has a hidden field that holds a frame number...
 put first word of field "Frame" into FNumber
 if field "Frame" is not empty then
 searchVideo FNumber —xcmd to find that frame on disc
 end if
end if
end openCard

on closeCard
 put empty into field "scrollBox"
end closeCard

```

on goToCard
  global Index, theList
  if theList is EMPTY then
    go to card Index
  else
    go to card ID (first word of line index of theList)
  end if
end goToCard

on mouseStillDown
  —This is used only by the scroll bar to allow for continuous scroll
  send mouseDown to the Target
end mouseStillDown

on InitializeScrollBar
  —Written by Luc Perron, 1988, RIT
  —This part of the code is to define the global variables that
  —are going to be used for the scroll bar. It should be used every
  —time there is a change in the total number of items in theList
  —The scroll box contains 3 buttons and one field.
  global maxLeft, maxRight, deltaX, V
  global theList, index, indexMax, theStep
  put 1 into index
  if theStep is empty then put 10 into theStep
  if theList is empty then
    put the number of cards into indexMax
  else
    put the number of lines in theList into indexMax
  end if
  get rect of field "scrollBox"
  put (third item of it) - (first item of it) into width
  —save the y coordinate (V) so that the scrollbox
  —always move at the same height across the screen...
  put second item of the location of field "scrollBox" into V
  put first item of rect of bkgnd button "pageScroll" + (width div 2)~
  into maxLeft
  get rect of bkgnd button "Next"
  put third item of rect of bkgnd button "pageScroll" - (width div 2)~
  into maxRight
  put (maxRight - maxLeft) / indexMax into DeltaX
end InitializeScrollBar

on PlaceScrollBar
  —Written by Luc Perron, 1988, RIT
  —This part will place the scroll box at its proper place each time
  —you open a new card. (More calculations...)
  global MaxLeft, MaxRight, DeltaX, V, Index, indexMax, theList
  put Index into field "scrollBox"
  if Index = 1 then
    put MaxLeft into H — This is the place for the first card
  else if Index = indexMax then
    put MaxRight into H — This is the place for the last card
  else
    put trunc (MaxLeft + (Index - 1) * DeltaX + DeltaX / 2) into H
  end if
  show field "scrollBox" at H, V — Show the box at its place
end PlaceScrollBar

```

on DragBox

- Originally written by Luc Perron
- Modified to work with large stacks 6/10/88 MAC/SHK
- This code is to allow the user to drag the scroll box around while displaying the proper card number, and then go to the selected card as soon as the user releases the mouse button.

```
global MaxLeft, MaxRight, DeltaX, V, Index, theList, IndexMax
repeat until the mouse is up
  put the mouseH into H
  if H < MaxLeft then put MaxLeft into H
  else if H > MaxRight then put MaxRight into H
  show field "scrollBox" at H, V
  add 0.49 to H
  put trunc ((H - MaxLeft - 0.5) / DeltaX) + 1 into temp
  if temp < 1 then
    put 1 into Index
  else
    put temp into index
  end if
  put index into field "scrollBox"
end repeat
goToCard
end DragBox
```

on ScrollPage

- Written by Luc Perron
- This code controls the action when the user clicks in the black area of the scroll bar. If user clicks on the right of the scroll box, go forward, if user clicks on the left, go backward.
- At the moment, it does the same as the arrow buttons, but the code could easily be modified to move by more than one step.
- Put the increment desired into "theStep" in initializeScrollBox.

```
global Index, theStep, indexMax
if the mouseH > first item of location of field "scrollBox" then
  add theStep to Index
  visual effect scroll left —slow
else
  visual effect scroll right —slow
  subtract theStep from Index
end if
if Index > indexMax then put indexMax into Index
else if Index < 1 then put 1 into Index
goToCard
end ScrollPage
```

on ScrollLeft

- Written by Luc Perron
- Go to the previous card in the list.

```
global Index, indexMax
visual effect scroll right
if Index > 1 then
  subtract 1 from Index
  goToCard
end if
—wait 90 ticks
end ScrollLeft
```

```

on ScrollRight
  —Written by Luc Perron
  —Go to the Next card in the list.
  global index, indexMax
  visual effect scroll left
  if index < indexMax then
    add 1 to index
    goToCard
  end if
  — wait 90 ticks
end ScrollRight

on sort
  —The cards have been sorted to put them in specific order...
  answer "It would not be wise to sort this stack" with "OK"
end sort

on doMenu command
  if command = "Delete Card" then
    beep
    —answer "Please don't delete any cards!" with "OK"
  else
    if command = "QuitHyperCard" then —""Change this
      —To insure a fresh start...

      if the freesize of this stack > 0.15 * the size of this stack then
        doMenu "compact stack"
      end if
      go to cd Main of stack "Beall Archive"
      pass doMenu
    end if
  end if
  pass doMenu
end doMenu

** BACKGROUND #1: Cards *****
—Hidden fld "End
—Hidden fld "Frame"
—Hidden bg btn "Detail Filler"

on closecard
  if visible of bg fld "Top Bar" then hideButtons
end closecard

on hideButtons
  lock screen
  hide bg fld "Top Bar"
  hide bg btn "Put Away" —Put away box
  hide bg btn "Intro.Nav"
  hide bg btn "Bio.Nav"
  hide bg btn "Query.Nav"
  hide bg btn "Time Line.Nav"
  hide bg btn "Data Cards.Nav"
  hide bg btn "Main.Nav"
  hide bg btn "Rectangle"
  hide bg btn "Cover2"

```

```

unlock screen
end hideButtons

on openCard
if bg fld "Details" is not empty then
hide bg btn "Cover"
put 1 Into bg fld "Slider"
show bg fld "Slider" at 45,321
else if bg fld "Details" is empty then
show bg btn "Cover"
end if
pase opencard
end openCard

on idle
showpict "GDA LOGO.2", 15,9
showpict "Lester Beall", 97,8
showpict "Data Cards", 96,31
if visible of bg fld "Top Ber" then
showpict "Navigator.ICON.2", 18,252
else
showpict "Navigator.ICON", 18,252
end if
pass idle
end idle
** BKGND #1, FIELD #20: Scrolling *****
on closeField
updateField the short name of me
hide me
openCard
end closeField

** BKGND #1, FIELD #22 *****
on mouseDown
-- Taken from "101 Scripts and Buttons
-- Compute the Minimum and Maximum X values from the Scrollbar rect.
-- Grab and preserve starting point, then enter the repeat loop as
-- long as the mouse is still down.
--
put left of bg btn "Scrollbar" + 10 into MinX
put right of bg btn "Scrollbar" - 10 into MaxX
put loc of me into newLoc
repeat while the mouse is down
--
-- Test to see if the mouse is less than the maximum X.
-- If so, then proceed, otherwise set loc of elevator to
-- maximum setting.
--
if the mouseH < (MaxX+1) then
--
-- Test to see if the mouse is less than the minimum X.
-- If so, then set loc of elevator to minimum setting.
-- If not, then set loc of elevator to mouse's X position.
--
if the mouseH < MinX then
put MinX into item 1 of newLoc
set loc of me to newLoc
else

```

```

    put the mouseH into item 1 of newLoc
    set loc of me to newLoc
end if
—
— Set the value in the display field
—
setit
else
—
— Set elevator to maximum value
—
put MaxX into item 1 of newLoc
set loc of me to newLoc
—
— Set the value in the display field
—
setit
end if
end repeat
lock screen
put bg fld "Slider" into LineNo
put line LineNo of bg fld "Details" into FrameNo
searchVideo FrameNo
unlock screen with visual effect wipe left
end mouseDown

on setit
—
— Compute the Minimum X and Difference X values from the Scrollbar.
— Grab the maximum numeric value from the field.
—
put left of bg btn "Scrollbar" into MinX
put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 10→
into DifX
put number of lines of bg fld "Details" into MaxVal
—
— Compute the correct setting for the displayed value
—
put round ((the mouseH - MinX)/DifX*MaxVal) into holder
if holder ≤ 1 then put 1 into holder
if holder ≥ MaxVal then put MaxVal into holder
put holder into line 1 of bg fld "slider"
end setit
** BKGND #1, FIELD #23: scrollBox *****
on mouseDown
  DragBox
end mouseDown

** BKGND #1, FIELD #24: Top Bar *****
on mouseStillDown
  global diffx,diffy
  put item 1 of the mouseLoc into x
  put item 2 of the mouseLoc into y
  if y > 60 and x > 100 then
    show me at x-diffx,y-diffy
    show bg btn "Rectangle" at ((x-diffx)+1),(y-diffy)+128)
  end if
end mouseStillDown

```

```

on mousedown
  global diffx,diffy
  put item 1 of the loc of me into CenterPtX
  put item 2 of the loc of me into CenterPtY
  put item 1 of the mouseLoc into x
  put item 2 of the mouseLoc into y
  put (x-CenterPtX) into diffx
  put (y-CenterPtY) into diffy
end mousedown

on mouseUp
  put item 1 of the loc of me into x
  put item 2 of the loc of me into y
  lock screen
  show bg btn "Put Away" at x-102,y+5 —Put away box
  show bg btn "Intro.Nav" at x+0,y+38
  show bg btn "Bio.Nav" at x+92,y+106
  show bg btn "Query.Nav" at x+60,y+210
  show bg btn "Time Line.Nav" at x-59,y+210
  show bg btn "Data Cards.Nav" at x-92, y+106
  show bg btn "Main.Nav" at x+0,y+135
  unlock screen
  showpict "Navigator", x-113,y-6
end mouseUp
-- BKGND!#1, BUTTON #2: LeftArrow -----
on mouseDown
  — Taken from "101 Scripts and Buttons"
  — Compute the Minimum, Maximum and Difference X values using the
  — Scrollbar's rect, then enter the repeat loop as
  — long as the mouse is still down.
  —
  put left of bg btn "Scrollbar" + 10 into MinX
  put right of bg btn "Scrollbar" - 10 into MaxX
  put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 24 →
  into DifX
  put one into MinPos
  put number of lines of bg fld "Details" into MaxPos
  repeat while the mouse = down
  —
  — Determine next value for display field, then set the loc of
  — the elevator to the correct X position for that value.
  —
  put bg fld "slider" - 1 into NextPos
  if NextPos ≤ MinPos then
    set the loc of bg fld "slider" to MinX, →
    item 2 of loc of bg fld "slider"
    put "1" into bg fld "slider"
  else
    put NextPos into line 1 of bg fld "slider"
    set loc of bg fld "slider" →
    to round ((NextPos/MaxPos) * DifX + MinX), →
    item 2 of loc of bg fld "slider"
  end if
  put bg fld "Slider" into LineNo
  put line LineNo of bg fld "Details" into FrameNo
  searchVideo FrameNo
end repeat

```

```

end mouseDown
** BKGND!#1, BUTTON #4: RightArrow *****
on mouseDown
  — Taken from "101 Scripts and Buttons
  — Compute the Minimum, Maximum and Difference X values using the
  — Scrollbar's rect, then enter the repeat loop as
  — long as the mouse is still down.
  —
  put left of bg btn "Scrollbar" + 10 into MinX
  put right of bg btn "Scrollbar" - 10 into MaxX
  put right of bg btn "Scrollbar" - left of bg btn "Scrollbar" - 24-
  into DifX
  put one into MinPos
  put number of lines of bg fld "Details" into MaxPos
  repeat while the mouse = down
    —
    — Determine next value for display field, then set the loc of
    — the elevator to the correct X position for that value.
    —
    put bg fld "slider" + 1 into NextPos
    if NextPos ≥ MaxPos
      then
        put MaxPos into bg fld "slider"
        set loc of bg fld "slider" to MaxX,-
        item 2 of loc of bg fld "slider"
      else
        put NextPos into bg fld "slider"
        set loc of bg fld "slider" -
        to round ((NextPos/MaxPos) * DifX + MinX),-
        item 2 of loc of bg fld "slider"
      end if
    put bg fld "Slider" into LineNo
    put line LineNo of bg fld "Details" into FrameNo
    searchVideo FrameNo
  end repeat
end mouseDown
** BKGND!#1, BUTTON #6: pageScroll *****
on mouseDown
  —see script of this stack
  ScrollPage
end mouseDown

** BKGND!#1, BUTTON #8: Previous *****
on mouseDown
  —see script of this stack...
  ScrollLeft
end mouseDown

** BKGND!#1, BUTTON #10: Next *****
on mouseDown
  —see script of this stack...
  ScrollRight
end mouseDown

** BKGND!#1, BUTTON #11: Return *****
on mouseUp
  pop card

```



```

end mouseUp

** BKGND!#1, BUTTON #13: Compass *****
on mouseUp
  if not visible of bg fld "Top Bar" then
    show bg fld "Top Bar" at 301,60
    put item 1 of the loc of bg fld "Top Bar" into x
    put item 2 of the loc of bg fld "Top Bar" into y
    lock screen
    show bg btn "Cover2"
    show bg btn "Rectangle" at 302,188
    show bg btn "Put Away" at x-102,y+5 ---Put away box
    show bg btn "Intro.Nav" at x+0,y+38
    show bg btn "Bio.Nav" at x+92,y+106
    show bg btn "Query.Nav" at x+60,y+210
    show bg btn "Time Line.Nav" at x-59,y+210
    show bg btn "Data Cards.Nav" at x-92, y+106
    show bg btn "Main.Nav" at x+0,y+135
    unlock screen
    showpict "Navigator", x-113,y-6
  else if visible of bg fld "Top Bar" then
    HideButtons
  end if
end mouseUp

** BKGND!#1, BUTTON #14: Put Away *****
on mouseup
  lock screen
  hide me
  HideButtons
end mouseup

** BKGND!#1, BUTTON #16: Intro.Nav *****
on mouseUp
  hideButtons
  go to cd Intro of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #17: Bio.Nav *****
on mouseUp
  lock screen
  hideButtons
  unlock screen
  go to cd Bio of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #18: Query.Nav *****
on mouseUp
  hideButtons
  go to stack "Quotes"
end mouseUp

** BKGND!#1, BUTTON #19: Time Line.Nav *****
on mouseUp
  push this cd
  hideButtons
  go to stack "Time Line"
end mouseUp

```

```

** BKGND!#1, BUTTON #20: Data Cards.Nav *****
on mouseUp
  push this cd
  hideButtons
  go to stack "Data Cards"
end mouseUp

** BKGND!#1, BUTTON #21: Main.Nav *****
on mouseUp
  hideButtons
  go to cd Main of stack "Beall Archive"
end mouseUp

** BKGND!#1, BUTTON #22 *****
on mouseUp
  if not visible of fld "Details" then
    show bg fld "Details"
  else
    hide fld "Details"
  end if
end mouseUp

** BKGND!#1, BUTTON #23: Detail Filler *****
on mouseUp
  set cursor to watch

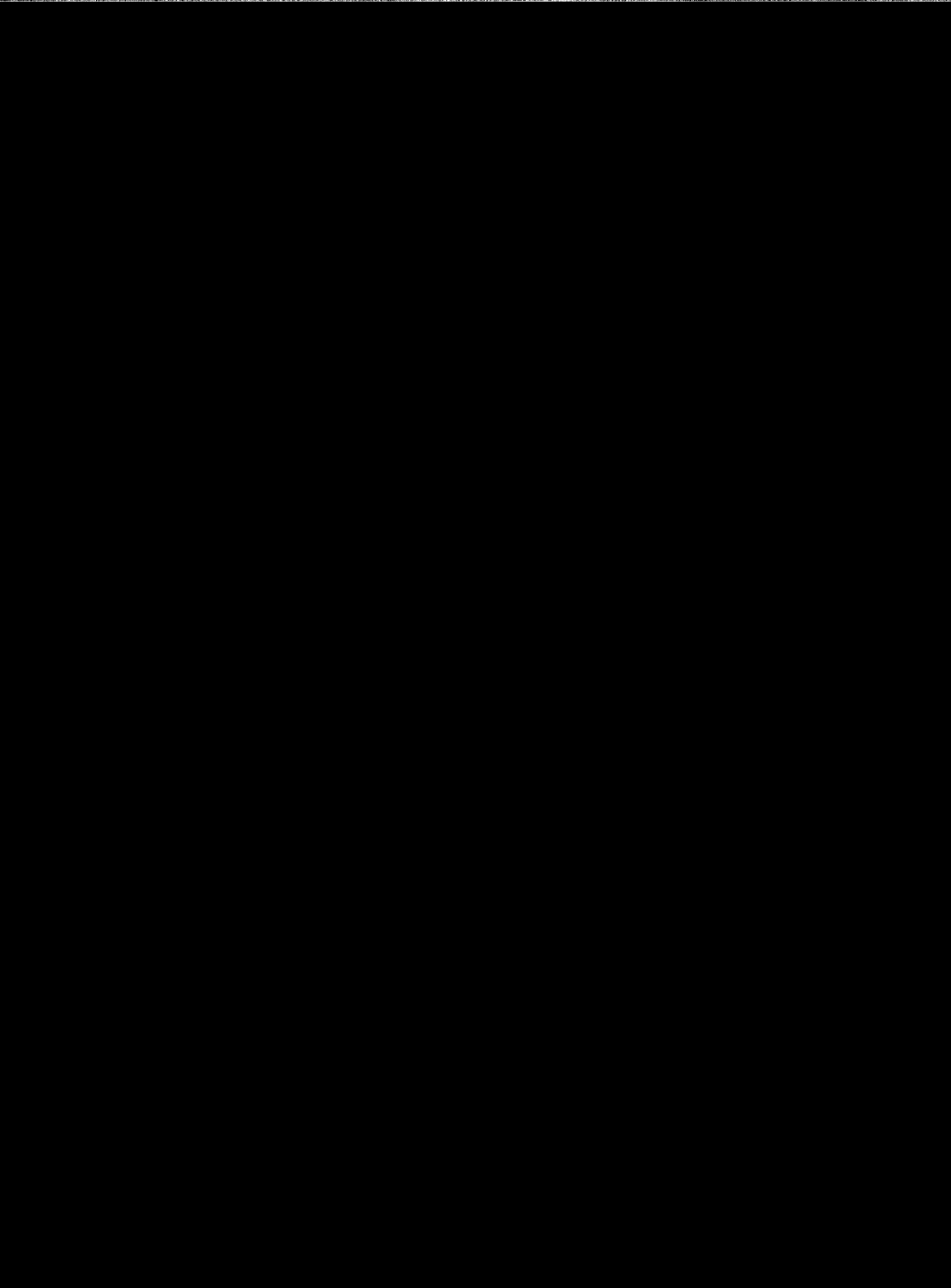
  put bg fld "Frame" into x
  put bg fld "End" into y

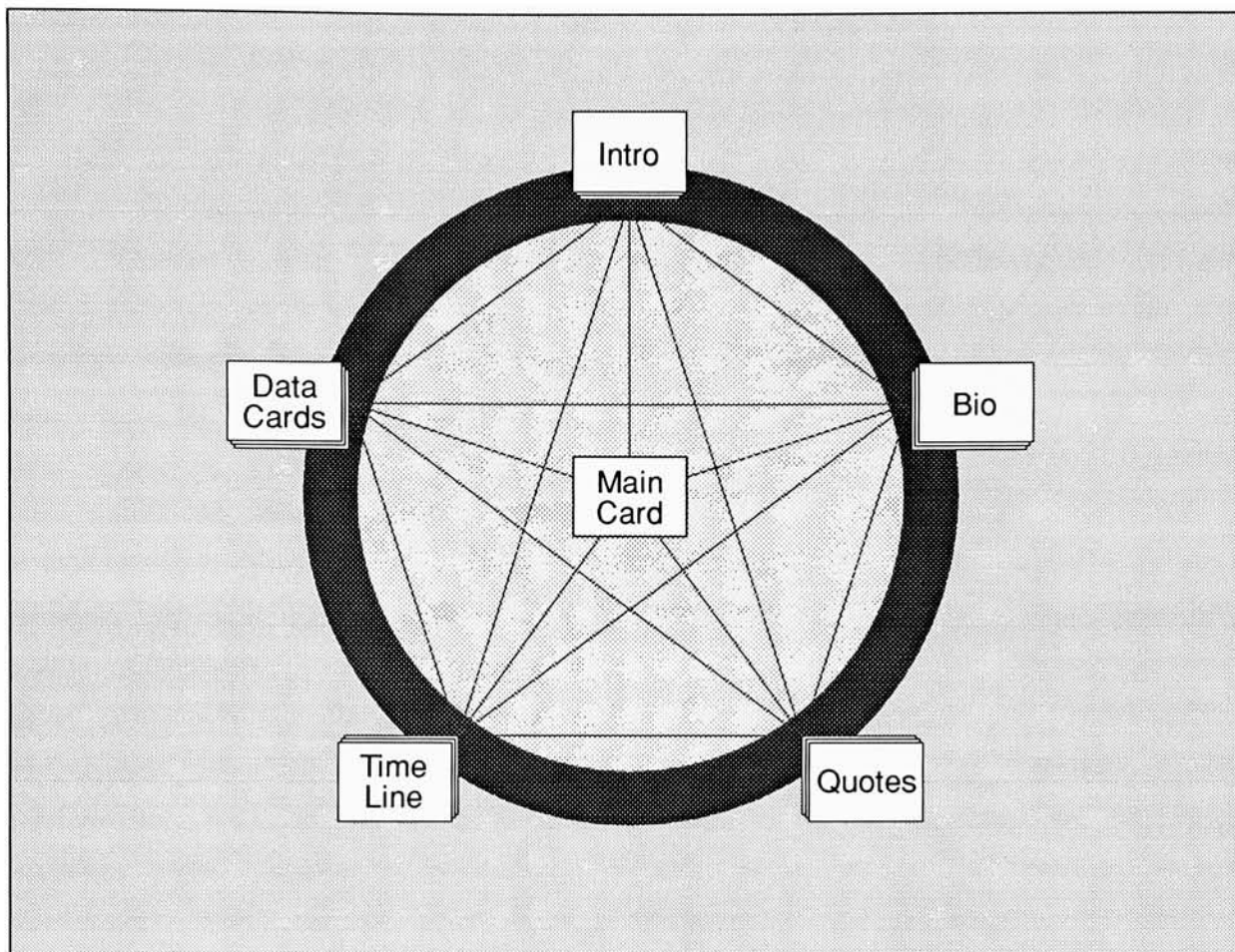
  put x into bg fld "Details"
  repeat with counter = x+1 to y
    put return&"0"&counter after bg fld "Details"
  end repeat
end mouseUp

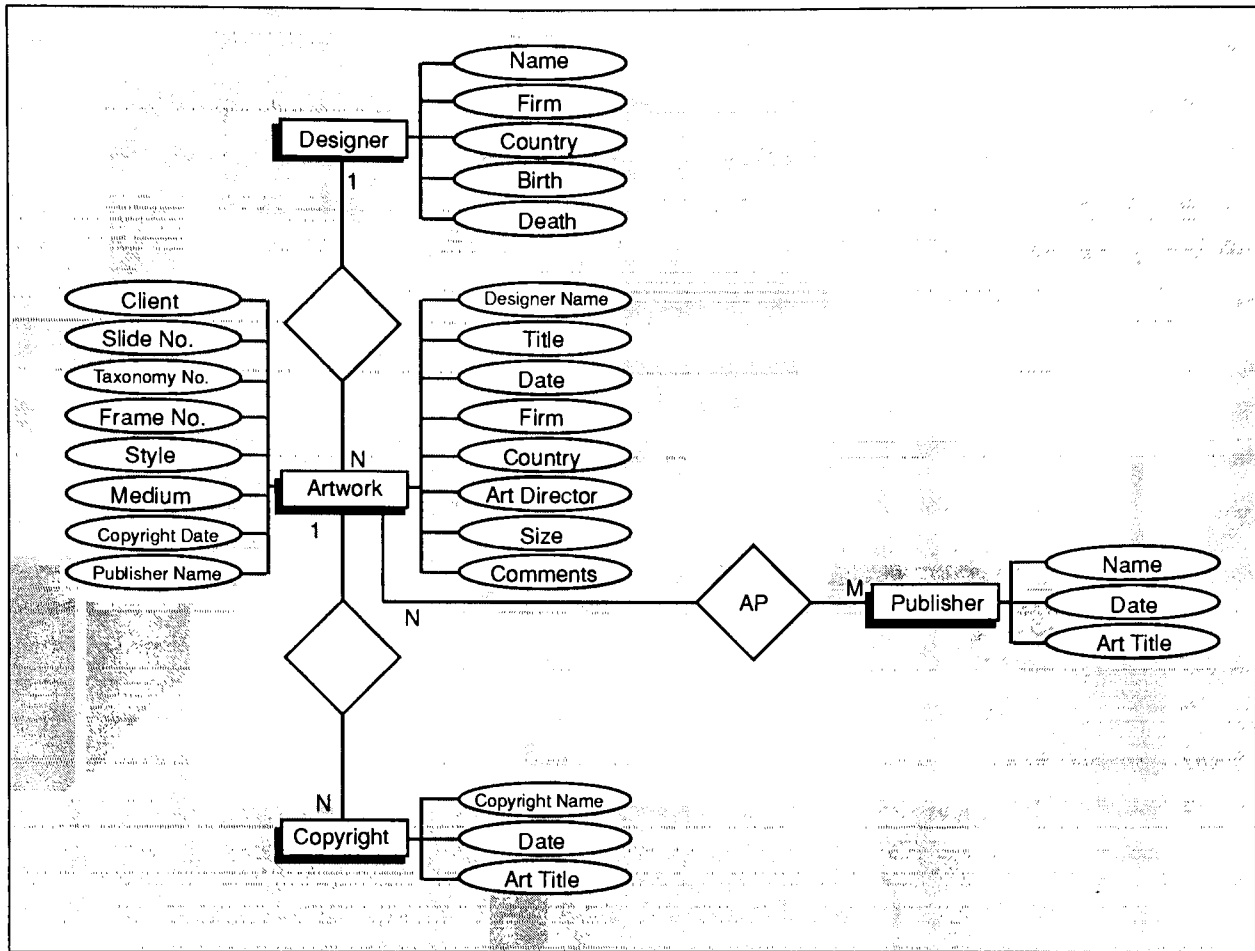
** BKGND!#1, BUTTON #24: Cover *****
on mouseUp

end mouseUp

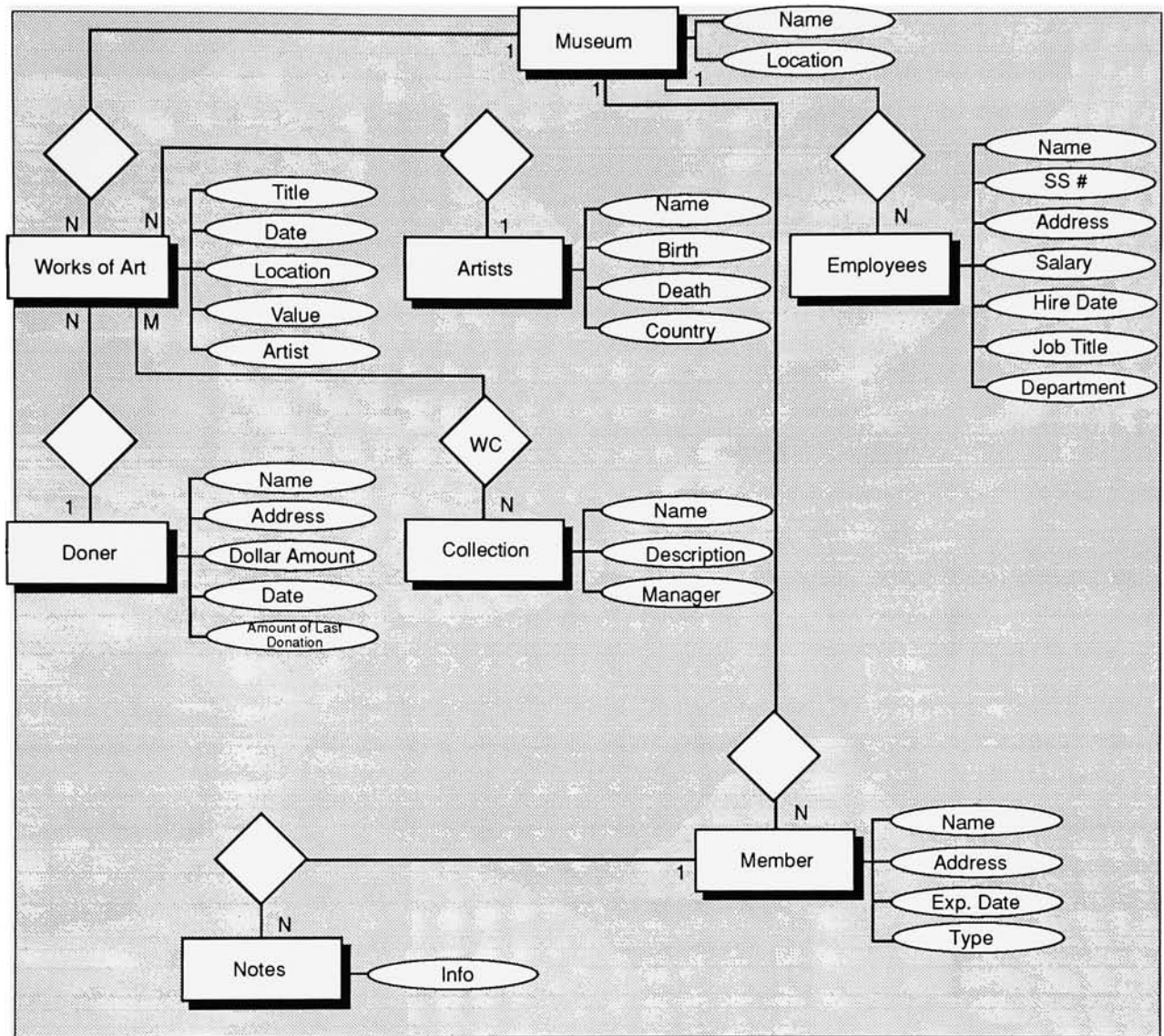
```





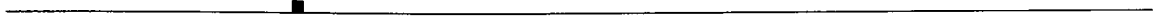


- Designer (Name, Firm County, Birth/Death, Artwork Title)
- Artwork (Designer Name, Title, Date, Firm, Country, Art Director, Size, Comments, Client, Slide No., Taxonomy no., Frame No., Style, Medium, Copyright Date)
- Copyright (Title of Copyright, Date, Artwork Title)
- AP (Publisher Name, Artwork Title)
- Publisher (Name, Date, Artwork Title)



- Museum (Museum Name, Museum Location)
- Works of Art (Title, Date, Location, Value, ArtistName, Museum Name, Doner, Collection Name)
- Doner (Doner Name, Address, Dollar Amount, Date, Amount of Last Donation)
- WC (Title, Collection Name,)
- Collection (Collection Name, Description, Manager)
- Artists (Artist Name, Birth, Death, Country)
- Employees (Emp. Name, SS#, Address, Salary, Hire Date, Job Title, Department)
- Member (Member Name, Member Address, Exp.Date, Type, Notes)
- Notes (Info, Member Name)

Supporting Papers/Correspondences



Prototype 3
David LaMarca
October 26, 1989

Draft1

Objective:

Develop a system of handling the images on disk with their accompanying detail shots. We must decide how to handle these details on the data cards.

Improve the database capabilities, this may involve going to another software application. We could either look at applications that would work with Hypercard, like HyBase and Oracle, or porting over to a new application all together, like SuperCard, Plus, or the IRIS Intermedia package.

Allow for dynamic linking by the user and the ability to save "webs."

Allow the user, especially the "expert designer" to be able to add expert commentary to the archive.

Improve the ability to save image sets, possibly setting up a custom scrapbook, with thumbnail and table of contents capabilities.

Current User profile:

-
- the serious researcher, writer, editor, and historian
 - the teacher
 - the remote teacher, or student
 - the average or advanced student
 - specialized museum applications
 - the professional designer
 - the casual browser
-

Current functions:

-
- Archival Functions
 - increased data capabilities
 - detail shots of artwork
 - finding guide
 - scan function
 - expert commentary
 - Supportive Tools
 - glossary of identity terms
 - bibliography
 - subject classification
 - style
 - time line (possibly dicotic?)
 - General Functions
 - more linkages of text and image
 - print out capacity - text and image
 - query atop query
 - functional slide show
 - mini authoring system for custom presentation
 - ability to generate own stack
 - camera tool
 - recent or preview (map system)
-

Prototype 3
David LaMarca
December 14, 1989

Draft 2

Update:

This is a statement of the things which I hope to accomplish on the prototype 3 project. I hope to bring many of the working tools from prototype 2 into this new prototype. Changes will of course be made to relate both visually and functionally to the new prototype. I will be developing a new interface for the prototype, along with an extensive navigational system. The prototype will be centered around Lester Beall as the main subject matter. It is my hope that this will be a modular system, where this prototype will just be one node in a series of other nodes, that when combined, will make a total archive.

Prototype 3
David LaMarca
January 11, 1990

Draft 3

User descriptions:

High end --	Library Installation
	Educators
	Students
	Library Personnel
Low end --	Researchers
	Professionals

Tools:

- Help System
- Navigational System
- Recent or preview window to tie in with Navigational system
- Query System
- Tree Diagram -- ?
- Quotes
- Camera -- ?
- Note pad to write down thoughts and possible grab little thumb nails of images
- Time Line
- The ability to layout images, similar to laying out a slide show traditionally
- Scanning Ability

Graphic Design Archive

Software Prototype 3

1. Develop interface to handle master images with details.
2. Expanded textual and other references, perhaps timelines, charts, etc.
3. Improve database capabilities.
4. Extend current linking to include additional text and other material, and expand within current Ref. stack.
5. Allow for dynamic linking by the user and the ability to save "webs".
6. Improve the capability to save image sets and edit them, i.e. Camera tool of some sort.
7. Data entry!

Problem Statement

The Graphic Design Archive is a desk-top information resource about the history of graphic design. It brings together the image storage capacity of the laserdisc with the data and interactive storage capabilities of the computer.

It users will range from the casual browser to the expert researcher. It will be found in design offices, libraries and archives, faculty offices, and in offices of graphic design advocacy organizations. While it may be used for verbal and visual information about many designers and their work, it may also focus in depth on the archive on one or more designers.

The current prototype, number 2.0, allows the user the interact with 800 images by 20 designers with access by designer, location and medium. Other tools include a help section, text information stack, a query function, scroll-bar function and a classification tree.

The problem at hand is to conceptualize new ideas relative to the function, scope, direction and operations that will enhance the short term and long term development of the project.

Print

Prototype #3 Development

	USER PROFILES	FUNCTIONS of archive
Primary	-Serious Researcher Writer Editor Historian -Teacher -Specialized Museum application -Remote Teacher/student -Professional Designer -Advanced Student	1. Link Text and Image 2. Print-out capacity(Text and Video) 3. Be able to generate your own stack in succinct manner and be able to save 4. Query on top of query 5. <u>Special Archival Functions</u> : -More Data -Details (close ups) -Finding guide -Orientation to Beall Archive 6. Camera Tool, List 7. Key Words(Glossary of terms) 8. Help Info for Identity Mark Classification Terms 9. Mini-Authoring system for customized presentation 10. More Linkages 11. Time Line 12. Style 13. Subject Classification 14. Recent or preview 15. Have functional "slide show" capacity 16. Bibliography 17. Expand "Expert" commentary capacity
Other	-Student -Casual Browser	

	USER PROFILES	FUNCTIONS of archive
Primary	<ul style="list-style-type: none"> -Serious Researcher Writer Editor Historian -Teacher -Specialized Museum Application -Remote teacher/Student -Professional Designer -Advanced student -Student 	<ol style="list-style-type: none"> 1. <u>Archival Functions</u> <ul style="list-style-type: none"> -Increased Data -Details(close ups) - <i>MAG. GLASS</i> -Finding guide -Orientation tool to Beall Archive - <i>SCAN</i> -Allow for expert commentary - <i>OPEN ARCHITECTURE DESIGN</i> - <i>EXPERT COMMENTARY</i> - <i>TIME LINE</i> 2. <u>Supportive Tools</u> <ul style="list-style-type: none"> -glossary of Identity Terms -Bibliography -Subject Classification -Style -Time Line -Key Words - <i>NOTE PAD</i> - <i>MAP - (RECENT WINDOW LIKE)</i>
Other	<ul style="list-style-type: none"> -Casual Browser 	<ol style="list-style-type: none"> 3. <u>General functions</u> <ul style="list-style-type: none"> -More linkages of text and image -Print our capacity-text & video -Query atop query -Functional slide show -Mini-Authoring System for custom present. -Be able to generate your own stack in succinct manner and save -Camera Tool -Recent or preview

TIED IN



Graphic Design Archive

Suggestions for Content and Format of Cataloging Records

Barbara Polowy, November 2, 1989

The following suggestions for the GDA record format are adapted from standard library cataloging rules and methods as set down in *Anglo American Cataloging Rules* (2nd ed.), MARC (Machine Readable Cataloging) format, and several special cataloging manuals used by the Library of Congress Prints and Photographs Division.

Library cataloging records adhere to a fairly rigid format in an effort to facilitate sharing of the information they contain through bibliographic utilities-- huge international databases. In addition to the elements in the record, the format determines the order of elements, the way names are recorded, and includes spacing and punctuation conventions to distinguish and identify the elements.

Since long-range plans for the GDA include pooling the records of several institutions, and since many of the graphic design archival collections are housed in libraries, adaptation of some or all of these established practices may facilitate development of a sharable resource.

I. Location

- A. Actual item -- institution, identifying number (call number, box number, item number, accession number, etc.)
- B. Reproduction in GDA--frame number?

II. Title of the item

●Comments

The title could be recorded directly from the item itself or a descriptive title supplied by the cataloger (usually distinguished from a title taken from the item itself by enclosure in brackets [])

III. Responsibility for creation of the item

- A. Personal names--eg. graphic designer, illustrator, type designer or letterer, photographer, art director, client (if an individual)
- B. Corporate names--eg. client (if a corporation), design firm or studio

●Comments

-Distinguishing between primary responsibility ("main entry") and secondary responsibility ("added entries") can be a time consuming task and such decisions can often be very subjective. Making these distinctions should not be mandatory if all names are equally accessible in the database

-Including the function served by each person and corporate body involved in the creation of the item would facilitate the generation of useful indexes to the database

-Maintaining a name authority file based on standard methods of name heading formation such as those described in *Anglo American Cataloging Rules*, 2nd ed. (or using an established thesaurus of names such as the Library of Congress Name Authorities) is of paramount importance to ensure successful retrieval of information from the database using names as search terms. The use of standard forms of names will be especially important if several institutions contribute to the database. Some considerations in forming names are:

GDA-Suggestions for Content and Format of Cataloging Records--2

- Choice of name
 - *Use of pseudonyms, well-known nicknames, or actual given name (eg. El Lissitzky, Eliezer Lissitzky, or Lasar Markowitz Lissitzky)
 - *Fullness--use of initials, middle names, titles (eg. Dr. Agha, M.F. Agha, or Mehemed Fehmy Agha)
 - *Language -- important consideration for geographic names (eg. Wien or Vienna, München or Munich)
- Order of name elements in the record (natural or inverted)
- Use of qualifiers to distinguish persons, bodies, or places with the same names--life dates, occupations, extra geographical elements (for places)
- Linking used and other forms of the names to ensure users retrieve material regardless of the name form used as a search term--i.e. providing see and see also references

IV. Circumstances of the items's creation

- A. Places of publication, distribution, and/or manufacture (if a multiple)
- B. Publishers, distributors, and/or manufacturers (if a multiple)
- C. Date of publication or creation--eg. printing, copyright, etc.
 - If no date information is available on the item, the cataloger should try to establish a probable date to the nearest decade if possible and indicate the approximation on the record by a question mark

V. Physical description

- A. Extent--number of physical units
- B. Description of what the item is--standard terms are given in the *Anglo American Cataloging Rules* (art original, poster, etc.), the Library of Congress Prints and Photographs Division uses a more extensive list, or a special list could be compiled for this project
- C. Medium--eg. engraving, lithograph, pencil drawing
- D. Color--limited to 3 or 4 major tones
- E. Dimensions
 - Including and excluding mounts, frames, etc.
 - In centimeters, height, width, depth

VI. Other Information

- A. Language
- B. Donor, source, previous owners of the item
- C. Variations in the title, other title information
- D. Source of the information used in the description if not from the item itself (eg. reference works, periodical articles, personal communications with the maker, donor, etc.)
- E. Edition and history of the item
- F. Summary of the content of the item
- G. Supplementary technical information--eg. typefaces used in the piece
- H. Physical condition of the item (damage, eg.)

●Comments

This information could be in informal note form or specific fields could be provided for the cataloger. While all the information might be useful to the serious researcher, not all fields need to be indexed.

VII. Intellectual Content

- A. Style--major historical styles?(De Stijl, Bauhaus), descriptive? (abstract, figurative, geometric, biomorphic)
- B. Subjects--what's shown? (light bulbs, ocean liners)
- C. Function--intended use? (trademark, magazine cover, package)

●Comments

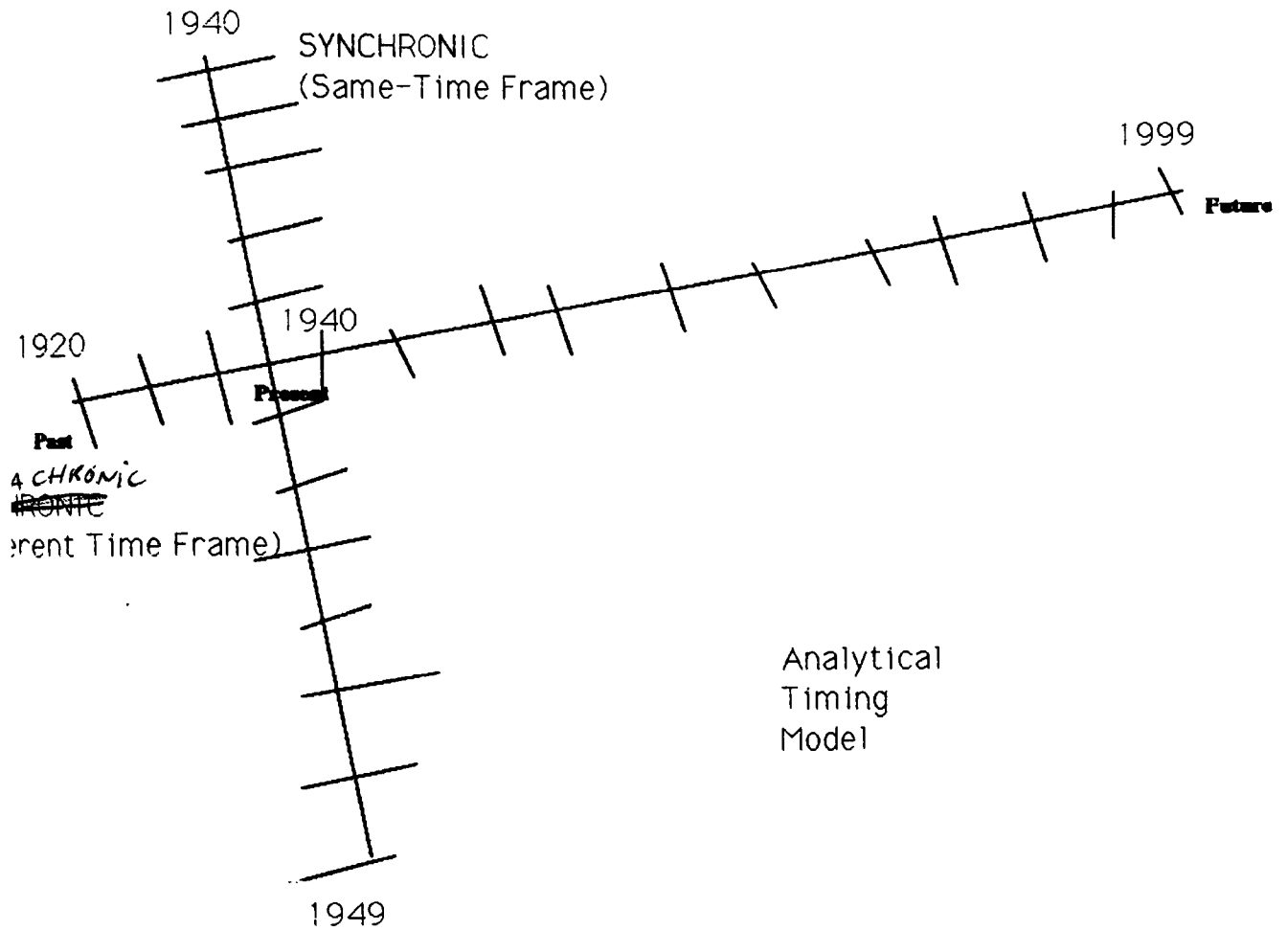
This area is the one which presents the most problems for both cataloging and retrieval. Terms used in this area could be assigned by the cataloger from his/her own natural vocabulary or chosen from a restricted list of terms (a controlled vocabulary in the form of a thesaurus or subject heading list) developed by experts in the field of graphic design. Both methods have numerous advantages and disadvantages. Ultimately the success of subject retrieval depends upon how well the terms used by the cataloger to describe the item match the terms used by the researcher trying to find information in the database. The importance content as an access point to information about graphic design (as opposed to names, places, material formats, etc.) should be the determining factor in whether or not the considerable time and effort required to develop and maintain a controlled vocabulary is part of the GDA project.

VIII. Cataloging Information

- A. Cataloger
- B. Date record added to the database

APPROACHES TO ORGANIZING CONTENT

1. SEQUENTIAL - chronological approach to movements, artists, designers
2. RETROSPECTIVE - treats topics from the present to the past, stressing what is contemporary as related to its roots in the past.
3. THEORY DEVELOPMENT - deals with the development of graphic design philosophies as a component of design history
4. STYLE DEVELOPMENT - focuses on the factors involved in the birth and maturation of various design and art styles.
5. THEMATIC - traces historically how particular themes are treated during various time periods
6. MICRO/MACRO - examines specific designers and/or works that embody characteristics of a movement and which may provide generalizations applicable to other designers and/or works.
7. PROBLEM SOLVING - approaches aspects of art and design history from the aspect of necessity resulting in solutions
8. MEDIA - traces historically how the use of a particular media develops throughout various eras and movements.
9. PERSONS/PROCESSES/PRODUCTS - examines the interrelationships of persons, their environment and their products.



R.I.T.
Wallace Memorial Library
David LaMarca
January 15, 1990

Macintosh Work Station Proposal

Workstation:

System Unit	Retail Price
Macintosh IIcx	\$ 7069.00
Apple RGB Color Monitor	\$ 999.00
Apple 8 Bit Video Card	\$ 648.00
Apple Extended Keyboard	\$229.00
Apple CD SC Rom Player	\$ 1199.00
Laserwriter IINT	\$ 4999.00

Purpose:

The main purpose of this work station will be to house the Lester Beall Archive, and later a much larger graphic design archive. The program which will run on this workstation is currently being developed by David LaMarca, a graduate student in the School of Fine and Applied Arts here at R.I.T. The work is being done as a thesis on the part of Mr. LaMarca and is expected to be completed by March 30, 1990.

The workstation will be used as both a finding guide for the actual archive, as well as an extensive research tool. The program will be filled with much information about the life, work, and significance of Lester Beall.

F Y I

INPUT FROM COOPER-UNION'S
JIM WALDRON

4 Software Development

It is our strong belief that the NGDA should be more useful than an electronic picture book. Incorporating the historical and textual information described earlier in this paper will necessitate some reorganization of the software in order to seamlessly integrate the visual and written information. Adding features, such as text and digital output, would allow for a more versatile tool in the study and presentation of graphic design history. An outline follows;

Develop textual information topics

A larger base of historical information, as discussed earlier, will be chosen for inclusion. The information must be linked to a selection of pictorial information and a simple user interface worked into the stacks.

Integrate archival and textual information

Combine the data cards and reference stack into a single launching point. Further associate the relationship between reference card data and videodisc images. This would allow a user interested in a broader reference topic to view a selection of that topic without leaving the reference stack.

Expand current stack features

Adding reference points to the query function would utilize the broader topics available in the reference stack. Allow combination of topics generated through queries. Append Stack Builder to allow for presentation notes and link development between selected works.

Develop new features

Ease of collecting and compiling information from the NGDA for educational research and presentation is important for the end use of the archive. A text, and if possible, image export feature should be added to the software to facilitate easy transmission of data to research papers, and lectures. In addition, an integrated ability to develop stacks for use outside of the archive should be included. The archive should be the basis of other interactive presentations such as student and professional lectures given apart from the NGDA stacks.

The Cooper Union would enjoy teaming up with RIT to structure and implement the interface design refinements.

4 Software Development

It is our strong belief that the NGDA should be more useful than an electronic picture book. Incorporating the historical and textual information described earlier in this paper will necessitate some reorganization of the software in order to seamlessly integrate the visual and written information. Adding features, such as text and digital output, would allow for a more versatile tool in the study and presentation of graphic design history. An outline follows:

Develop textual information topics

A larger base of historical information, as discussed earlier, will be chosen for inclusion. The information must be linked to a selection of pictorial information and a simple user interface worked into the stacks. OK

Integrate archival and textual information

? Combine the data cards and reference stack into a single launching point. Further associate the relationship between reference card data and videodisc images. This would allow a user interested in a broader reference topic to view a selection of that topic without leaving the reference stack.

Expand current stack features

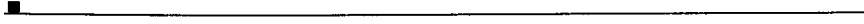
Adding reference points to the query function would utilize the broader topics available in the reference stack. Allow combination of topics generated through queries. Append Stack Builder to allow for presentation notes and link development between selected works. OK

Develop new features

Ease of collecting and compiling information from the NGDA for educational research and presentation is important for the end use of the archive. A text, and if possible, image export feature should be added to the software to facilitate easy transmission of data to research papers, and lectures. In addition, an integrated ability to develop stacks for use outside of the archive should be included. The archive should be the basis of other interactive presentations such as student and professional lectures given apart from the NGDA stacks. OK ?

The Cooper Union would enjoy teaming up with RIT to structure and implement the interface design refinements.

Synecotics Session on the GDA



SYNECTICS

Roger Remington

Alex Young

December 12, 1989

Group process - synectics session

Brainstorming of a type

seeking collective wisdom for long & short term goals.

ROGER is the juggler -

ground rules for session - want to push ourselves as far as we can - stretch our thinking

No value judgements until appropriate time

stages in a process.

* insights into new directions

group problem solving process

1st STEP (Roger as Client)

Problem statement:

WISH LIST

① Talk to it

show me you stuff

my own terms

② more portable

simpler set up

③ see ~~how~~ thought process of designer

④ Base search on what you have

⑤ access on phone.

⑥ speculate on social, political & environmental!

effects on his work

⑦ select images & extract them - slide show

on a given topic

⑧ random function then base search on what you choose

⑨ Talk back to user

⑩ Tactile output

⑪ contemporary designers (show)

⑫ talk to Bealls contemps

⑬ one screen system

⑭ publishable hard copy

of slide show you generate

SYNECTICS

Roger Remington

Alex Young

12/12/89

- ⑬ clip art tool box
- ⑭ connections 1
- ⑮ readily available to students
- ⑯ Comments by users to be recorded
- ⑰ output - laser printout or something

~~Thought~~ Thought process of designer

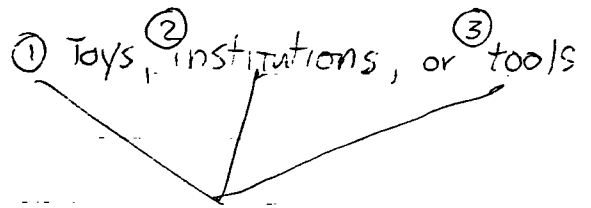
search from what you have already

contemporary designers past vs. present

Talk to it & have it talk back

EXCURSION - deal w/ lateral thinking

Take some of these ideas



metaphoric connection

slide show - play viewmaster

Transformers - change with the situation

clipart - Clay - Shape it

baseball card - reconfigure for special uses or nostalgic experiences

pictionary

Legos - modular

Tinker Toys

crawl explore pathways - DISCOVER

Institutional

Talk to it - get something back - process would be credible

Verox - maze (decentralized)

Synectics

Alex Young

Paper Remington

12/12/89

Institutions

① Telephone system as an index ④ Boss support system for his/her work

② Structure - RIT 18 years in the black

③ Catholic church

① chalk line plum line

② Post it notes

③ multiple use

Next Stage

Force Fit back through the metaphors & see how they fit to the problem statement

① interchangeability for diff. level of user

② open ended system - click in new data Legos

- "GDA" central based on simple clear premises

- find one good thing that works & stay with it - legos

③ USER / LEARNER how do people learn - need for more definition
cast 4 learning types against what we are doing here so that
what we do can apply to all.

④ Designers thoughts post it notes → process of designer
documentation of ideas

⑤ impact all the senses (RELATES TO # 3)

⑥ connectedness - explore ideas, context

⑦ output - what you take away - hard copy

⑧ lester respond to user expert system

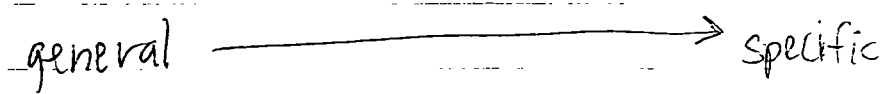
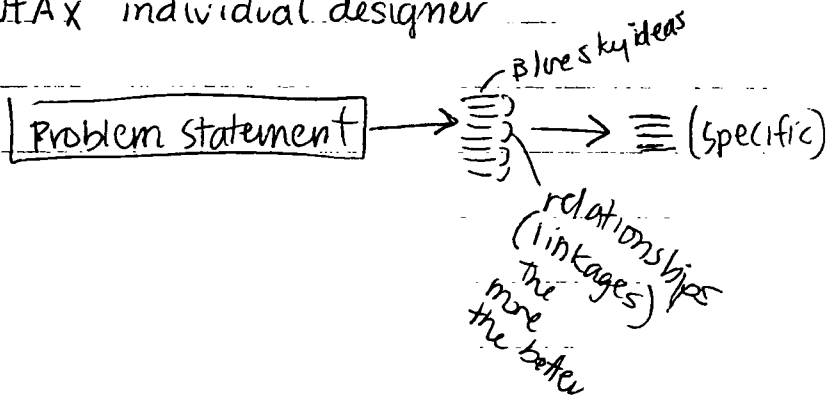
Honesty

SYNECTICS

Roger Remington

Alex Young
12/12/89

- Expert valuation
- LINEARITY - sequences
- Hyper multi-media interactive choices
- GDA Scripting levels
- SYNTAX individual designer



Prototype 2.0 Screen Designs





Graphic Design Archive

Introduction

The focus of the prototype
is the following group of designers
who worked between 1930 and 1955:

Saul Bass	E. McKnight Kauffer
Herbert Bayer	Gyorgy Kepes
Lester Beall	Leo Lionni
Joseph Binder	Alvin Lustig
Alexey Brodovitch	Herbert Matter
Will Burtin	Paul Rand
A. M. Cassandre	Ladislav Sutnar
Charles Eames	Bradbury Thompson
William Golden	Jan Tschold

 **Return**
QUIT



Graphic Design Archive

Introduction

The Graphic Design Archive is being developed by:
Rochester Institute of Technology
One Lomb Memorial Drive
Rochester, New York U.S.A.

It is a cooperative effort of.
Department of Graphic Design
College of Fine & Applied Arts
American Video Institute
Department of Applied Computer Studies
College of Liberal Arts

 **Return**
QUIT



Graphic Design Archive

Introduction

Project Director: **R. Roger Remington**

Project Designer: **Cathleen Britt**

Director, American Video Institute: **Dr. John Ciampa**
Project Coordinator: **Mark Collien**
Project Archivist: **Sandra Markham**
Project Evaluator: **Susan Preston-Mauks**

◀▶ **Return**

QUIT



Graphic Design Archive

Introduction

Faculty / Staff Consultants:

Steven Kurtz
Chris Comte
Dr. Barbara Hodik
Robert Keough

Programmers:

Cathleen Britt Luc Perron
Steven Kurtz Ruth Cohen
Mark Collien

Student Development Assistants:

Jessica Loy Paul Getman
David LaMarca Bonnie House
Lynn Egger Susan Miller
Luo Perron John Malinoski
Ruth Cohen Catherine Elkin

◀▶ **Return**

QUIT



Graphic Design Archive

Introduction

Graphic Design Archive Fellows:

Jerry Counselman
Julie Stachowiak
Sr. Alma Mary Anderson C.S.C.

Reference Stack Concept Development

Ellen Lupton, Curator
The Herb Lubalin Study Center
of Design and Typography
The Cooper Union

◀ ▶ **Return**
QUIT



Graphic Design Archive

Introduction

The project would not have been possible without the generous support of Apple Computer, Inc. and Digital Equipment Corporation. In addition, the following agencies/groups have funded portions of the Graphic Design Archive:

National Endowment for the Arts
New York State Council on the Arts
Graham Foundation
Department of Graphic Design, RIT
College of Fine & Applied Arts, RIT
American Video Institute, RIT

◀ ▶ **Return**
QUIT



Stack Builder

The Stack		Images
Designer		Drawn/painted form 1
Location		USA 730
Medium		Printed image 98
File		
Tree		
Query		

Scan <> Save Remove

Ref. Cards

START QUIT



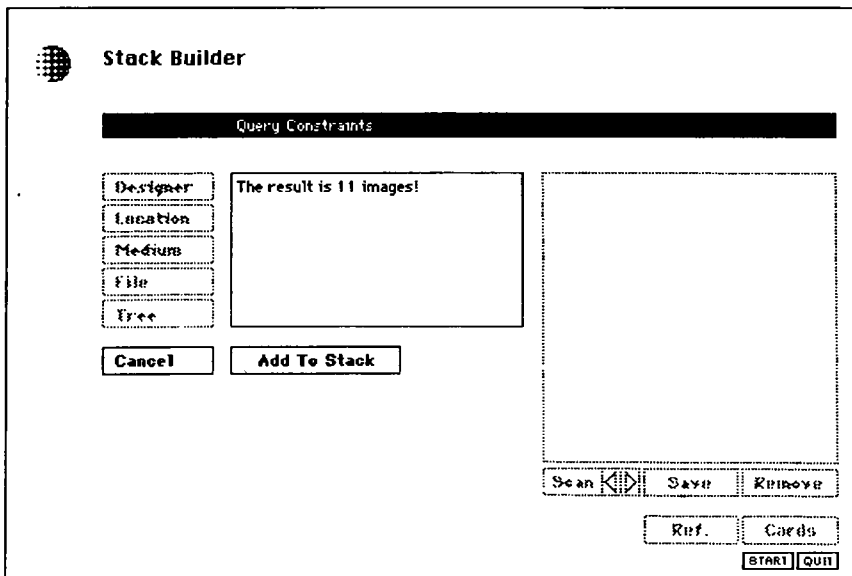
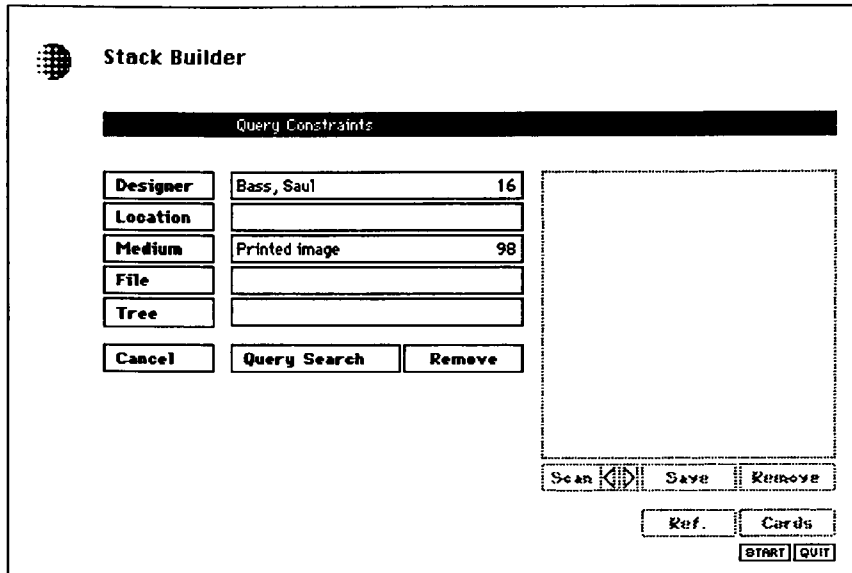
Stack Builder

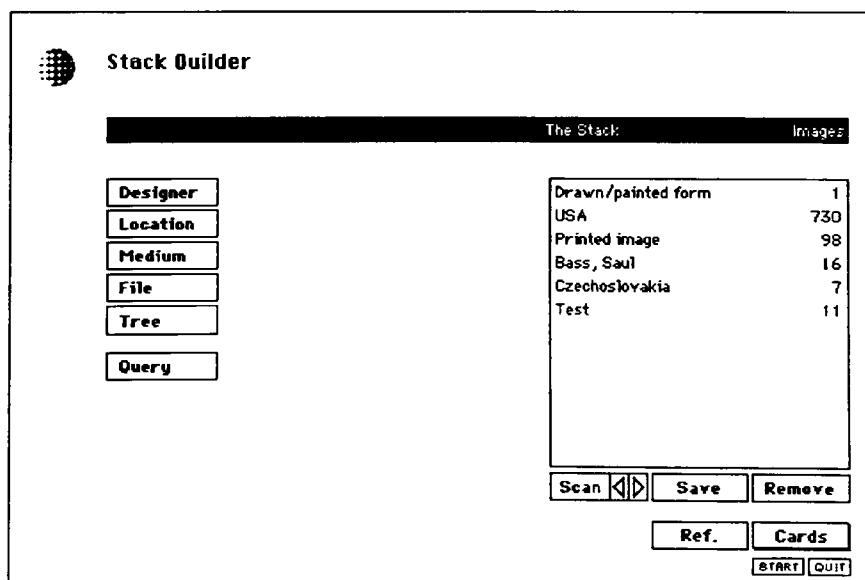
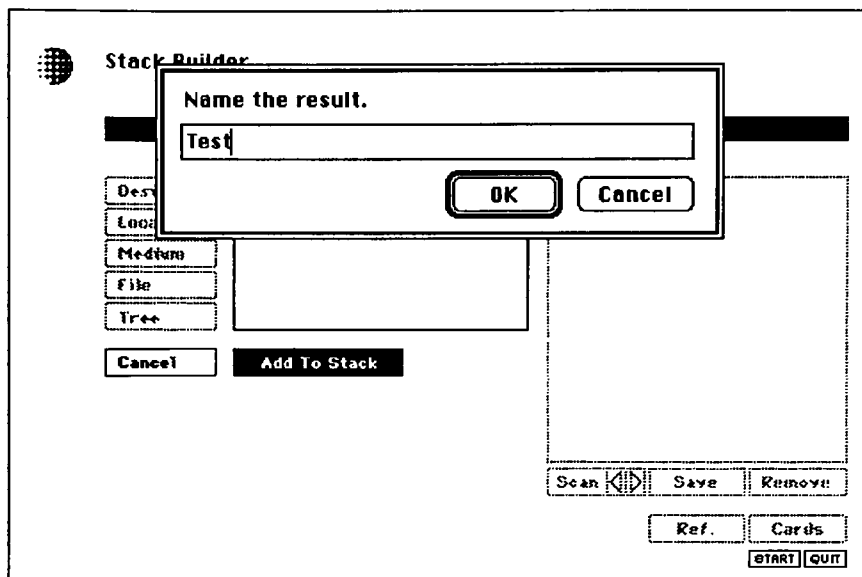
The Stack		Images
Designer	Bass, Saul	Drawn/painted form 1
Location	Bayer, Herbert	USA 730
Medium	Beall, Lester	Printed image 98
File	Binder, Joseph	
Tree	Brodovitch, Alexey	
Query	Burtin, Will	
	Cassandre, A. Mouron	
	Eames, Charles	
	Golden, William	
	Guisti, George	
	Kauffer, E. McKnight	
	Kepes, Gyorgy	


Scan <> Save Remove

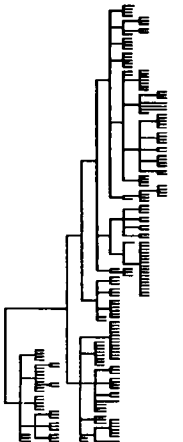
Ref. Cards

START QUIT






 **Classification Tree**



Classification Tree | **Design Work**
804 images | **Designers Archive**

Scan <> **Return**

START **QUIT**

 **Reference**

Introduction | **See Also**

Click on the **Index** button to bring up a scrolling field of topics. Choose one of the topics by clicking on it.

Click on the **Notes** button to access bibliographic information.

Click on the **Return** button in the lower right corner of this card to get back to the place you came from.

Topics that are listed under "See Also" can be clicked on to access cards that contain information about them.

Index **Notes** **Return**

START **QUIT**



Reference

Introduction

See Also

Click on the **Index** button to bring up a scrolling field of topics. Choose one of the topics by clicking on it.

Click on the **Notes** button to access bibliographic information.

Click on the **Return** button in the lower right corner of this card to get back to the place you came from.

This is the place that bibliographical information can be found. Look in the "Notes" area on the other cards in this stack.

Index

Notes

Return

START **QUIT**



Reference

Introduction

See Also

Click on the **Index** button to bring up a scrolling field of topics. Choose one of the topics by clicking on it.

Click on the **Notes** button to access bibliographic information.

Click on the **Return** button in the lower right corner of this card to get back to the place you came from.

American School
Art Nouveau
Arts and Crafts Movement
Bass, Saul
Bauhaus
Bayer, Herbert
Beall, Lester
Binder, Joseph
Bradley, William
Brodovitch, Alexey
Burtin, Will
Cassandre, A. Mouron
Constructivism
Container Corp. of America
Cubism
de Stijl

Index

Notes

Return

START **QUIT**



Reference

Beall, Lester

See Also

Lester Beall (b.1903-d.1969), was educated at Lane Technical School and University of Chicago, although not in graphic design. He worked in Chicago, New York and Connecticut. He was honored in 1937 as the first graphic designer to have a one man show at the Museum of Modern Art in New York. His work is distinguished by his use of a unique mix of media and the obvious influences of Constructivism and the Bauhaus. Beall is recognized for his posters, corporate identity and other forms of graphics, as well as for his positive influence on the relationship between client and designer and his ideas concerning the effect of design on society. His major clients were International Paper, Connecticut General, Caterpillar and Upjohn.

**American School
Bauhaus
Constructivism**

Index

Notes

Return

START QUIT



Data Card

124

Designer	Cassandre, R. Mouron
Date	1937
Location	USR
Medium	Printed image, lithography

Title Magazine advertisement for Container Corporation of America, "Diversification"

Comment

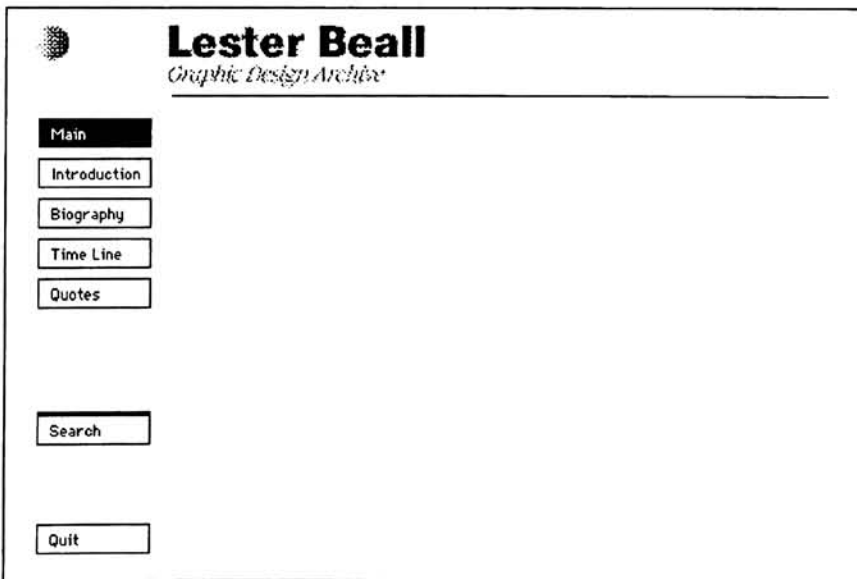
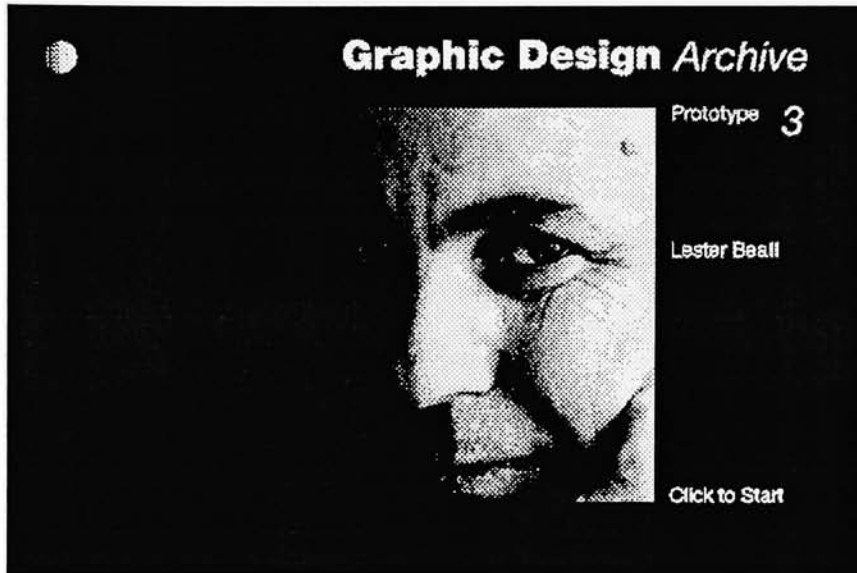
Ref.

Return

START QUIT

Prototype 3.0 Screen Designs







Lester Beall

Introduction

Main

Introduction

Biography

Time Line

Quotes

The purpose of this prototype of the Graphic Design Archive was to develop a working archive of Lester Beall's work, which was donated to the Rochester Institute of Technology by the Beall Family.



Lester Beall

Introduction

Main

Introduction


Biography

Time Line

Quotes

The theory behind this archive was to work off of a node system. A node is basically a subset of a much larger part. This design is closely related to fractal forms, where one node has several identical nodes attached to it. In turn each node also has several subsets and so on and so on.





Lester Beall

Introduction

Main

Introduction

Biography

Time Line

Quotes

The case with this archive is such that the Graphic Design Archive is the main section. This prototype is but just one of the nodes to be added to the main. Each stack in this prototype represents a node of it, and following suit so does each card represent a node of each stack. It is through this design structure that the Graphic Design Archive will be one large, stable, functioning unit

←

→

Lester Beall

Introduction

Main

Introduction

Biography

Time Line

Quotes

It is with hope that someday the Graphic Design Archive will be a collaboration of schools, archivists, and museums spanning the spectrum of graphic design, both past and present.

←

→



Lester Beall

Biographical information:

Main

Introduction

Biography

Time Line

Quotes

(1903 - 1969)

A pioneer of twentieth-century American graphic design,

Lester Thomas Beall was born in Kansas City, Missouri, on 14 March 1903; his childhood years were spent in St. Louis and Chicago. Beall was educated at Chicago's Lane Technical School and began his

design career in 1927, the year following his graduation from the University of Chicago. He married Dorothy Wells Miller in 1928 and they



were the parents of two children, Lester Beall, Jr., born in 1928, and Joanna May Beall, born in 1935.

By 1935 Beall had decided to move to New York and in September of that year opened a studio/office/apartment in Tudor City on Manhattan's east side. In 1936, while maintaining his office in New York, he moved to Wilton, Connecticut where he established his home



Lester Beall

Continued...

Main

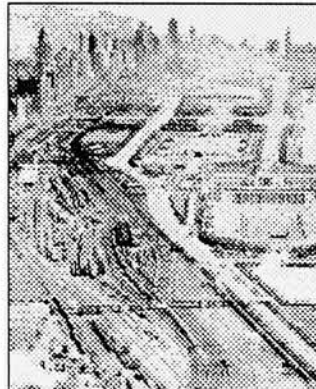
Introduction

Biography

Time Line

Quotes

and office in a rural setting. He was to remain in Wilton until 1950. Many of the significant work from this period were done in his studio in Wilton. Through the 1930s and 40s Beall produced innovative and highly regarded work for the Chicago Tribune, Sterling Engraving, The Art Directors Club of New York, Crowell Publishing Company, Hiram Walker, Abbott Laboratories, Time Magazine and the United States Government's Rural Electrification Administration, among others.



Chicago - 1930's





Lester Beall

Continued...

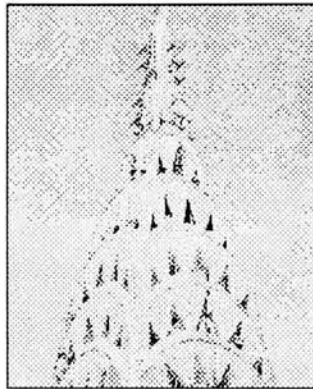
Main

Introduction

Biography

Time Line

Quotes



Chrysler Building, NYC -1940's

Beall had moved his office to 580 Fifth Avenue by 1946 and worked from there as well as from his rural Connecticut home and studio, Dumbarton Farm in Brookfield Center, which he had purchased in 1950. In 1952 Beall took an office at 60 Sutton Place in Manhattan; in 1955 he moved to consolidate all his operations at Dumbarton Farm, developing some of the working farm's outbuildings into professionally praised office and studio space.

During the 1950s and 60s Beall's design office expanded both its staff and scope, adding



Lester Beall

Continued...

Main

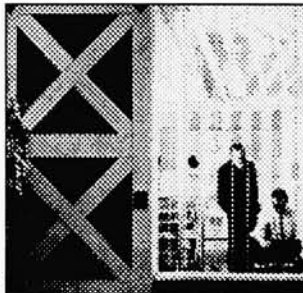
Introduction

Biography

Time Line

Quotes

associate designers and mounting full scale corporate identification campaigns for large companies such as International Paper, Caterpillar Tractor, Martin Marietta and Rohm and Haas. Beall incorporated his business in June of 1960 and it continued in operation until shortly after his death on June 20, 1969.



Throughout his career Lester Beall was a highly respected designer whose thoughts and principles made him a favored lecturer in professional and educational circles. He wrote, and was written about, extensively, and his

work was regularly featured in American and international design periodicals and books.

During his lifetime he participated in more than





Lester Beall

Continued...

Main

Introduction

Biography

Time Line

Quotes

one hundred exhibitions in the United States and abroad, including over ten one-man shows, and won over 150 professional awards for his graphic and package designs as well as his personal oil and watercolor paintings.

The Lester Beall Collection consists of an extensive written and visual collection documenting a lifetime of distinguished professional achievement. The correspondence, photographs, design samples, published and unpublished writings, biographical material and business papers were assembled by

Beall, his staff and, especially, his devoted wife of 41 years, Dorothy Miller Beall. It provides a remarkably comprehensive record of the designer and his life, and reveals every phase of the creative process which has established Lester Beall's contribution to the history of graphic design in America. ■



Lester Beall

Time Line

Client Info

Biographic

Historic

Quotes

	1920
1921	Warren G. Harding, President Klee joins Bauhaus
1922	Constructivism in Russia W.A. Dwiggins coins term 'graphic design' Moholy-Nagy does first experimental photographs First advertisement on radio
1923	George Grosz publishes Ecoe Homo Cassandre posters first appear in Paris
1924	Leica camera first produced



Lester Beall
ibmc Live

Client Info

Biographic

Historic

Quotes



1940

many graphic designers to utilize the sources of inspiration.

Date: 00/00/40
Author: Lester Beall
Title: "Modern Trends in Graphic Art"
Publication.
Page: 21

"The basic and the most richly endowed field of inspiration is nature itself. For example, a tree trunk might be the inspirational genesis for a photograph while its upper branches and leaves suggest the texture and color for the possible accompanying type form. The human form itself contains countless possibilities

Page 1 | Page 2

Lester Beall
ibmc Live

Client Info

Biographic

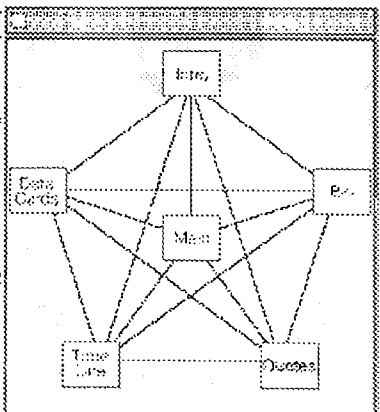


Historic

Quotes

1940
Client Name: Box
Title: Fever / Deh

1940
Client Name: Box
Title: ABC Christ

1940
Client Name: Box
Title: "Red Lion #



Lester Beall

Quotes

Main

Title: 'Design as applied to
Publication: American Printer

Date: 08/00/41
Page: 27

Introduction

Biography

Time Line

Quotes

'Modern graphic design is based on a consideration of the three dimensional, even though it is physioally two-dimensional. This seeming paradox can be clarified through an understanding of the functioning of spatlal planes in two-dimensional design. Planes in a design are established by oolor, form, and texture. Care must be taken so that no one of the planes becomes dissociated from the others, for properly related planes create a 'tension' that makes a good design 'work.'

Improperly related planes create confusion and a 'busyness' that contributes to the inefficiency of the advertisement.



Lester Beall

Graphic Design Archive

Title

Client

Medium

Subject

Box #


File

Cancel

Query Search

Remove


Quit

 **Lester Beall**
Graphic Design Archive

Title	Metal
Client	Mixed Media
Medium	Paint
Subject	Pen and Ink
Box#	Pencil
File	Photography
	Print

Cancel

Quit

 **Lester Beall**
Graphic Design Archive

Title		
Client		
Medium	Print	93
Subject		
Box#		
File		

Cancel **Query Search** **Remove**

Quit



Lester Beall

Graphic Design Archive

Title		
Client	Abbott Laboratories	4
Medium	Print	93
Subject		
Box #		
File		

Cancel	Query Search	Remove
--------	--------------	--------

Quit



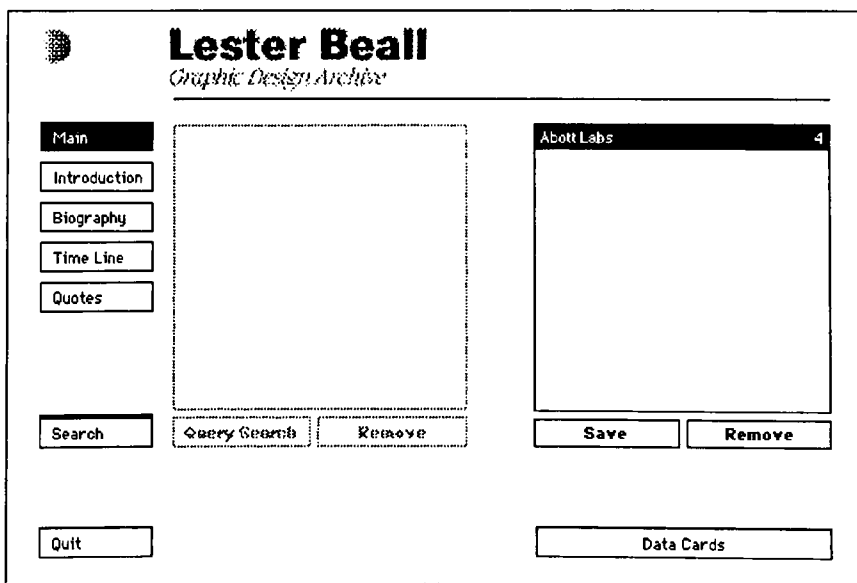
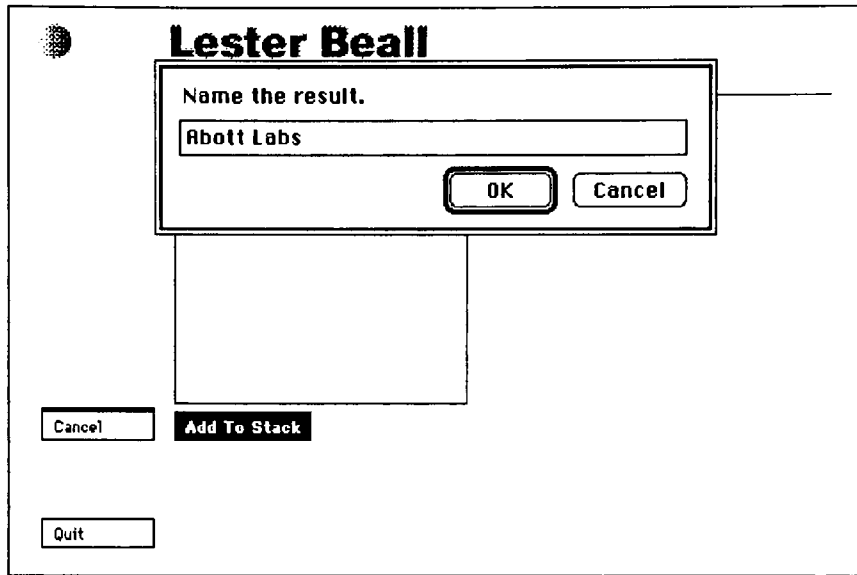
Lester Beall

Graphic Design Archive

The result is 4 images!

Cancel	Add To Stack
--------	--------------

Quit





Lester Beall

Brochures

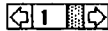
Date **1936** Location **Box 1**
 Title **ABD Vitamin Capsules Folder**
 Art Director
 Artist **Beall, Lester**
 Client **Abbott Laboratories**

Description
 Size (inches) **Width 12" Height 12" Depth 0"**
 Medium **Print**
 Subject **Brochure**

Copyright
 Publisher
 Source **Beall Collection** Date
 Comments Date



Details



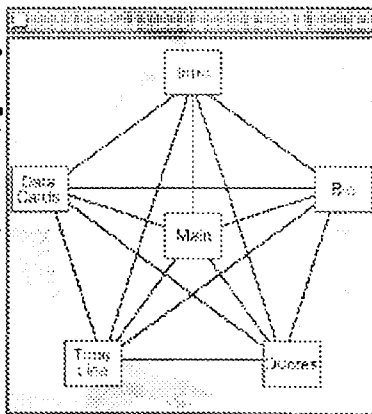
Lester Beall

Data Cards

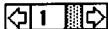
Date **1936** Location **Box 1**
 Title **ABD Vitamin Cap**
 Art Director
 Artist **Beall, Lester**
 Client **Abbott Laborato**

Description
 Size (inches) **Width 12"**
 Medium **Print**
 Subject **Brochure**

Copyright
 Publisher
 Source **Beall Collection** Date
 Comments Date



Details





Graphic Design Archive

Taxonomy

Designer

Artist

Art Director

Title/Description

Date Produced

Country of Origin

Size (in ~~cm~~)

Medium

Subject Class

Style

Client

Copyright ~~holder~~

Publisher

Source of Image

Comments

1AAAD

Frame # 08962

Location
Box # 25

Hofmann, Armin

NEW

Concrete Poetry

FIND

Earliest 19??

Latest 0

USA

W 32

H 24

D 0

Page 1

Printed Image, letterpress

SORT

Identity Mark

Abstract

QUIT

Date

Date

Strong mark, with heavy blacks and very bold negative space.

Much gestalt. Abstract symbolism is very evident.



Card by

John Doe

Date July 19, 1989



10050
10051
10052
10053
10054

separate

ADD :

1 WORD GRAPHIC

1 ILLUSTRATOR

1 TYPE DESIGNER

1 HISTORY OF PIECE

1 PRESERVATION CONDITION

1 SOURCE OF CATALOGUED INFORMATION



Graphic Design Archive

Taxonomy 1AAAD Frame # 08962 Location 25
Designer Hofmann, Armin
Artist _____
Art Director _____
Title Concrete Poetry
Description _____
Date 19??
Country of Origin USA
Size (inches) W 32" H 12" D 0" Page 1
Medium Printed Image, letterpress
Subject Class Identity Mark
Style Abstract
Client _____
Copyright Holder _____ Date _____
Publisher _____ Date _____
Source of Image _____
Comments Strong mark, with heavy blacks and very bold negative space.
Much gestalt. Abstract symbolism is very evident.

Card by John Doe Date July 19, 1989

NEW

FIND

SORT

QUIT





Graphic Design Archive

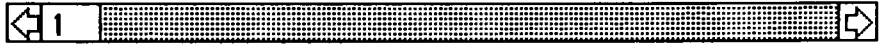
Taxonomy	<u>1AAAD</u>	Frame #	<u>08962</u>	Location	<u>25</u>	
Designer	<u>Hofmann, Armin</u>					
Artist					
Art Director					
Title	<u>Concrete Poetry</u>					
Description					
Date	<u>19??</u>					
Country of Origin	<u>USA</u>					
Size (inches)	<u>W 32"</u>	<u>H 12"</u>	<u>D 0"</u>	Page	<u>1</u>	
Medium	<u>Printed Image, letterpress</u>					
Subject Class	<u>Identity Mark</u>					
Style	<u>Abstract</u>					
Client					
Copyright Holder				Date
Publisher				Date
Source of Image					
Comments	<u>Strong mark, with heavy blacks and very bold negative space.</u> <u>Much gestalt. Abstract symbolism is very evident.</u>					
Card by	<u>John Doe</u>			Date	<u>July 19, 1989</u>	

NEW

FIND

SORT

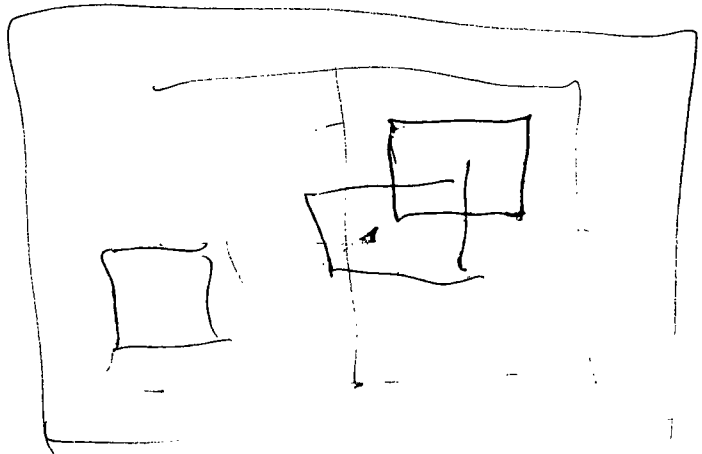
QUIT



- CATALOGER ALSO

- HAVE ~~THE~~ TYPIST LOG IN
AND HAVE ~~THE~~ SUBJECT NAME
GO INTO THE FIELD.

- ON NEW CARD, HAVE THE
ABILITY TO COPY CERTAIN
FIELDS INTO A NEW CARD.





Graphic Design Archive

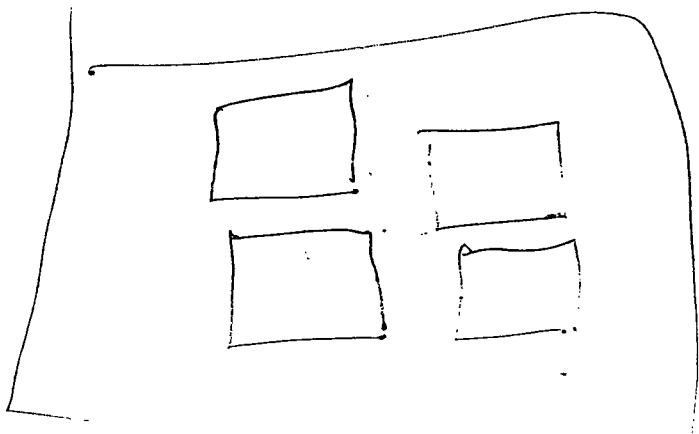
Taxonomy	1AAAD	Frame # 08962	Location 2.5	
Designer	Hofmann, Armin			FIND
Artist				SORT
Art Director				
Title	Concrete Poetry			
Description				
Date	19??			
Country of Origin	USA			
Size (inches)	W 3.2" H 1.2" D 0"	Page 1		
Medium	Printed image, letterpress			
Subject Class	Identity Mark			
Style	Abstract			
Client				
Copyright Holder		Date		
Publisher		Date		DETAIL
Source of Image				NEW
Comments	Strong mark, with heavy blacks and very bold negative space. Much gestalt. Abstract symbolism is very evident.			COPY
				QUIT
Cataloger Initials		Card By John Doe	Date 1/12/90	

can
all
fields

EVERY ON COMMENTS

ON RETURN

SET THE AUTO PDS





Lester Beall

SDA Prototype 3

(1903 - 1969)

A pioneer of twentieth-century American graphic design,

Lester Thomas Beall was born in Kansas City, Missouri, on 14

March 1903; his childhood years were spent in St. Louis and Chicago. Beall was edu-

cated at Chicago's Lane Technical School and began his

design career in 1927, the year following his graduation from the University of Chicago. He married Dorothy Wells Miller in 1928 and they



were the parents of two children, Lester Beall, Jr., born in 1928, and Joanna May Beall, born in 1935.

By 1935 Beall had decided to move to New York and in September of that year opened a studio/office/apartment in Tudor City on Man-

hattan's east side. In 1936, while maintaining his office in New York, he moved to Wilton, Connecticut where he established his home

--	--	--

--	--	--

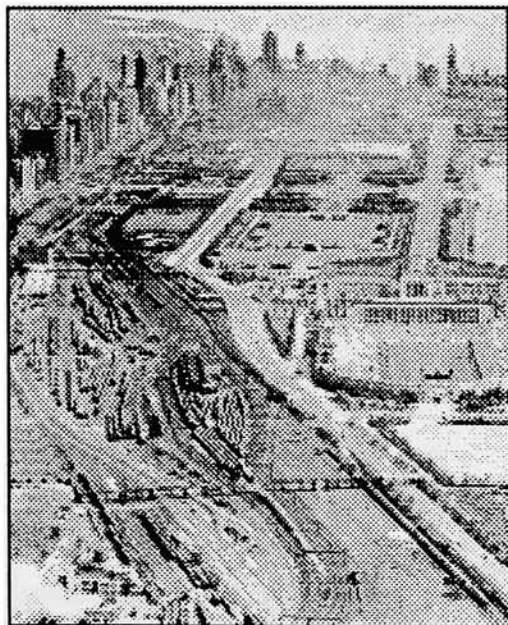


Lester Beall

SDA Prototype 3

and office in a rural setting. He was to remain in Wilton until 1950. Many of the significant work from this period were done in his studio in Wilton. Through the 1930s and 40s Beall produced innovative and highly regarded work for the Chicago Tribune, Sterling Engraving, The Art Directors Club of New York, Crowell Publishing Company, Hiram Walker, Abbott Laboratories, Time Magazine and the United States Government's Rural Electrification Administration, among others.

Beall had moved his office to 580 Fifth Avenue



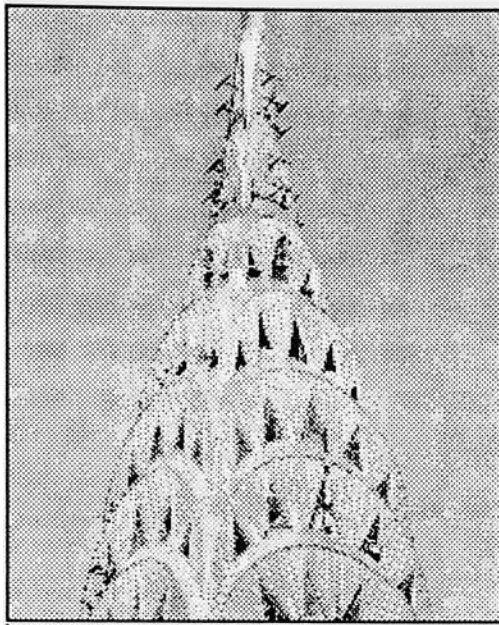
--	--	--

--	--	--



Lester Beall

GDA Prototype 3



--	--	--

by 1946 and worked from there as well as from his rural Connecticut home and studio, Dumbarton Farm in Brookfield Center, which he had purchased in 1950. In 1952 Beall took an office at 60 Sutton Place in Manhattan; in 1955 he moved to consolidate all his operations at Dumbarton Farm, developing some of the working farm's outbuildings into professionally praised office and studio space.

During the 1950s and 60s Beall's design office expanded both its staff and scope, adding associate designers and mounting full scale

--	--	--



Lester Beall

GDA Prototype 3

corporate identification campaigns for large companies such as International Paper, Caterpillar Tractor, Martin Marietta and Rohm and Haas. Beall incorporated his business in June of 1960 and it continued in operation until shortly after his death on June 20, 1969.

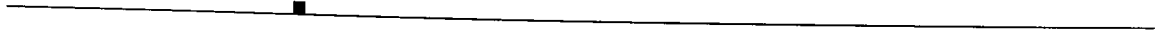
Throughout his career Lester Beall was a highly respected designer whose thoughts and principles made him a favored lecturer in professional and educational circles. He wrote, and was written about, extensively, and his work was regularly featured in American and

--	--	--

international design periodicals and books. During his lifetime he participated in more than one hundred exhibitions in the United States and abroad, including over ten one-man shows, and won over 150 professional awards for his graphic and package designs as well as his personal oil and watercolor paintings.

The Lester Beall Collection consists of an extensive written and visual collection documenting a lifetime of distinguished professional achievement. The correspondence, photographs, design samples, published and

--	--	--



The Apple Desktop Interface is the result of a great deal of concern with the human part of human-computer interaction. It has been designed explicitly to enhance the effectiveness of people. This approach has frequently been labeled *user friendly*, though *user centered* is probably more appropriate. It has been thought of as the ideal interface for beginners, though it would be more useful to think of it as good for people in general. It has been labeled *simple*, though *direct* and *effective* make more sense. And it has been described as *easy to learn*, though *accessible* would be as true.

A view of the user

Not very long ago, most users of personal computers were also programmers. In fact, many of them were computer builders as well, because personal computers were available only as kits. Today, most personal computers are seen as tools that magnify a person's ability to perform all kinds of tasks that were formerly done without computers. The Apple Desktop Interface provides a consistent and familiar computer environment in which people can perform their many tasks. People aren't trying to *use computers*—they're trying to get their jobs done.

Given this focus on people and their tasks, the Apple Desktop Interface has had to assume a model of people, in order to suit the interface to them. People, however, are delightfully complex and varied, which assures that a theory of human activity that would provide a complete framework for the design of human-computer interaction is a long way off. Such a theory would be oversimplified anyway, because computers themselves change the way we think, feel, and behave. Computer design and human activity must therefore evolve together. Apple believes that caring how people behave will help computer designers provide a consistent world that a person can enter with ease and effectiveness, even though many of the details of human activity are not understood.

The Apple Desktop Interface is based on the assumption that people are instinctively curious: they want to learn, and they learn best by active self-directed exploration of their environment. People strive to master their environment: they like to have a sense of control over what they are doing, to see and understand the results of their own actions. People are also skilled at manipulating symbolic representations: they love to communicate in verbal, visual, and gestural languages. Finally, people are both imaginative and artistic when they are provided with a comfortable context; they are most productive and effective when the environment in which they work and play is enjoyable and challenging.

Handwritten signature: Eric S. Raymond

8/11/82
Can do
with help
11.

The next section of this chapter sets forth ten human interface principles that explicitly emphasize the views expressed above. A subsequent section describes a programming strategy that takes these principles into account. Chapter 2 describes a specific set of elements for a Desktop Interface that can be used consistently across a range of very different applications. Chapter 3 provides standards and specifications for their implementation.

General design principles

This section describes the ten fundamental principles of the Apple Desktop Interface.

Metaphors from the real world

Use concrete metaphors and make them plain, so that users have a set of expectations to apply to computer environments.

Whenever appropriate, use audio and visual effects that support the metaphor.

Most people now using computers don't have years of experience with several different computer systems. What they do have is years of direct experience with their immediate world. To take advantage of this prior experience, computer designers frequently use metaphors for computer processes that correspond to the everyday world that people are comfortable with.

The desktop itself is the primary metaphor for the Apple Desktop Interface. It *appears* to be a surface on which users can keep tools and documents. Yet many of the elements of the Apple Desktop Interface don't have a clear physical counterpart. For example, scroll bars clearly belong to the computer domain; they only loosely resemble real scrolls. And pull-down menus aren't much like real restaurant menus, except in providing the opportunity to make choices from alternatives.

The desktop, then, is an inviting metaphor that provides easy access to the system. Other metaphors, especially when consistent with the desktop, can fit into the system. Once immersed in the desktop metaphor, users can adapt readily to loose connections with physical situations—the metaphor need not be taken to its logical extremes.

Direct manipulation

Users want to feel that they are in charge of the computer's activities.

People expect their physical actions to have physical results, and they want their tools to provide feedback. For example, when character keys are pressed, users like to hear a click as they see the corresponding characters appear on the screen. When a drawing tool is moved, a line appears. This is true whether or not a computer is being used. Moving a document from one folder or disk to another, or into the trash, can seem to be a physical activity with physical feedback in the computer world as it is in the paper world. The physical activity of moving the mouse reinforces the sense of an action with a real result.

Users want topics of interest to be highlighted. They want to see what functions are available at any given moment. If grave consequences might follow from any of those functions, they want to know about them—before any damage is done. They want clues that tell them that a particular command is being carried out, or, if it cannot be carried out, they want to know why not and what they can do instead.

People appreciate visual effects, such as animation, that show that a requested action is being carried out. This is why, when a window is closed, it appears to shrink into a folder or icon. Visual effects can also add entertainment and excitement to programs that might otherwise seem dull. Why shouldn't using a computer be fun?

See-and-point (instead of remember-and-type)

Users select actions from alternatives presented on the screen.

The general form of user actions is noun-then-verb, or "Hey, you—do this."

Users rely on recognition, not recall; they shouldn't have to remember anything the computer already knows.

Most programmers have no trouble working with a command-line interface that requires memorization and Boolean logic. The average user is not a programmer.

The Apple Desktop Interface is visually and spatially oriented. The way everything—text, applications, documents, lines, controls—appears on the screen is consistent and well thought out. The screen provides an environment in which people can work effectively, taking full advantage of the power of the computer while enjoying a sensible human environment.

Users interact directly with the screen, choosing objects and activities they are interested in by pointing at them. The mouse is currently the most common pointing device, but other effective pointing devices are available.

There are two fundamental paradigms for how the Apple Desktop Interface works. They share two basic assumptions: that users can see, on the screen, what they're doing; and that they can point at what they see. In one paradigm, users first select an object of interest (the noun) and then select an action (the verb) to be performed on the object. All actions available for the selected object are listed in the menus, so that users who are unsure of what to do next can quickly jog their memory by scanning through them. Users can choose, at any time, any available action, without having to remember any particular command or name. This paradigm requires only recognition, rather than recall, of the desired activities.

In the second paradigm, the user drags an object (the noun) onto some other object which has an action (the verb) associated with it. In the Finder, for example, the user can drag icons into the trash can, into folders, or into disks. No action is chosen from the menus, but it's obvious what happens to the object that is sent to another object. For example, an object sent to the trash can is discarded, and the document sent to a disk icon is copied to that disk. In this variant of the Desktop Interface, users do have to remember what an object such as the trash can is for, so it is especially important that objects look like what they do. If the trash can didn't look like the place to discard something, or we didn't know from daily experience that folders contain documents, such an interface wouldn't work. However, when this type of interface is well thought out, it can be easier to learn than menu commands.

Command-line interfaces, on the other hand, require the user to remember a command and type it into the computer. This kind of interface makes considerable demands on the user's memory—especially when the commands are complex or cryptic. Such an interface is especially galling to the new or infrequent user, but it distracts all users from their task and focuses attention instead on the computer's needs.

There are, however, some advantages to the *remember-and-type* approach. Sometimes, when the user is completely certain of what action is desired, a simple keystroke command may be the fastest way to achieve it. For this reason, some desktop applications include *keyboard equivalents* for some menu activities. Keyboard equivalents are a logical extension of the Apple Desktop Interface, fine-tuning it for particular situations. It is essential, however, that keyboard equivalents offer an *alternative* to the *see-and-point* approach—not a substitute for it. Users who are new to a particular application, or who are looking for potential actions in a confused moment, must always have the option of finding a desired object or action on the screen.

Consistency

Effective applications are both consistent within themselves and consistent with one another.

Having learned, in one application, a general set of skills, the user can transfer those skills to other applications. By using the standard elements of the Apple Desktop Interface, you ensure consistency within your application and you benefit from consistency across applications.

Within an application, there should always be one coherent way for the user to implement actions. Though some shortcuts may be provided, users should always be able to rely on familiar and straightforward ways to get things done.

The standard elements of the Apple Desktop Interface ensure consistency, ease of learning, and familiarity across applications. This benefits the typical user, who usually divides working time among several applications, and it benefits every software developer because the user learning how to use a new application builds on prior experiences with the same elements in other applications. This sometimes means that a programmer's new solution that precisely matches a particular situation should be set aside in favor of a slightly less effective but more commonly used solution. In most cases, consistency should be valued above idiosyncratic cleverness.

WYSIWYG (what you see is what you get)

There should be no secrets from the user, no abstract commands that only promise future results.

There should be no significant difference between what the user sees on the screen and what eventually gets printed.

A very important use of computers is the processing and printing of text and graphics. In some systems, the computer is an intermediary: the user manipulates a range of computer commands to indicate what is desired, and the computer passes these commands along to a printer. This kind of system keeps the user unnecessarily distant from the final document. The user should be in charge of both the content and the formatting (spatial layout as well as font choices) of the document. The computer should quickly and directly display the result of the user's choices, so the user doesn't have to wait for a printout or make mental calculations of how the screen version will be translated onto paper.

The principle behind this approach is known as *what you see is what you get* (abbreviated WYSIWYG, pronounced *witzzy-wig*). This approach is highly consistent with the direct manipulation aspect of the Apple Desktop Interface. For example, when a user uses the Finder to copy a document from one disk to another, the user "sees" a copy of the document move to the new disk and can trust that the document is now found on both disks. WYSIWYG is also in the spirit of using a computer as a thinking tool *and* as a production tool.

User control

The user, not the computer, initiates and controls all actions.

People learn best when they're actively engaged. Too often, however, the computer acts and the user merely reacts within a limited set of options. In other instances, the computer "takes care" of the user, offering only those alternatives that are judged "good" for the user or that "protect" the user from detailed deliberations.

On the surface, the concept of computer as protector may seem quite appealing, but this approach puts the computer, rather than the user, in the driving role—something quite at odds with the basic philosophy of the Apple Desktop Interface.

In the Apple Desktop Interface, if the user attempts something risky, the computer provides a warning, but allows the action to proceed if the user confirms that this is what he wants. This approach "protects" the beginner but allows the user to remain in control.

Feedback and dialog

Keep the user informed.

Provide immediate feedback.

User activities should be simple at any moment, though they may be complex taken together.

To be in charge, the user must be informed. When, for example, the user initiates an operation, immediate feedback confirms that the operation is being carried out, and (eventually) that it's finished. When the application isn't responding to user input because it's processing a different task, the user must be informed of when to expect delays, why, and for how long.

The user must also be kept informed of the progress of an operation: for example, the reason an operation can't be completed at a certain time as well as the fact that it can't.

This communication should be brief, direct, and expressed in the user's vocabulary rather than the programmer's.

Forgiveness

Users make mistakes; forgive them.

The user's actions are generally reversible—let users know about any that aren't.

Even though users like to have full documentation with their software, they don't like to read manuals (do you?). They would rather figure out how something works in the same way they learned to do things when they were children: by exploration, with lots of action and lots of feedback.

As a result, users sometimes make mistakes or explore a bit further than they really wanted to. Make your application tolerant and forgiving. Forgiveness means letting users do anything reasonable, letting them know they won't break anything, always warning them when they're entering risky territory, then allowing them either to back away gracefully or to plunge ahead, knowing exactly what the consequences are. Even actions that aren't particularly risky should be reversible. Tell the users about any exceptions to this rule.

When options are presented clearly and feedback is appropriate and timely, learning is relatively error-free. Alert messages should therefore be infrequent. If the user is subjected to a barrage of alert messages, something is wrong with the program design.

Perceived stability

Users feel comfortable in a computer environment that remains understandable and familiar rather than changing randomly.

People use computers because computers are versatile and fast. Computers can calculate, revise, display, and record many kinds of information far faster than people can. If users are to cope with the complexity that the computer handles so easily, they need some stable reference points.

To provide a visual sense of stability, the Apple Desktop Interface provides a two-dimensional space on which objects are placed. It also defines a number of consistent graphic elements (menu bar, window border, and so on) to maintain the illusion of stability.

To provide a conceptual sense of stability, the interface provides a clear finite set of objects and a clear finite set of actions to perform on those objects. Even when particular actions are unavailable, they are not eliminated from a display, but are merely dimmed.

It is the *illusion* of stability that is important, not stability in any strict physical sense. The environment can, and should, change as users interact with it, but users should feel that they have a number of familiar "landmarks" to count on.

Aesthetic Integrity

Visually confusing or unattractive displays detract from the effectiveness of human-computer interactions.

Different "things" look different on the screen.

Users should be able to control the superficial appearance of their computer workplaces—to display their own style and individuality.

Messes are acceptable only if the user makes them—applications aren't allowed this freedom.

In traditional applications, the visual appearance of the screen has been a low priority and consequently somewhat arbitrary. In contrast, the Apple Desktop Interface depends on the visual appearance of the screen. People deserve and appreciate attractive surroundings. Consistent visual communication is very powerful in delivering complex messages and opportunities simply, subtly, and directly.

Users should have some control over the look of their workspaces. This allows individual expression and relieves the computer designer of having to devise one "look" that appeals to everyone.

The next section summarizes some basic principles of visual design.

Principles of graphic communication

Good design must communicate, not just dazzle. It must inform, not just impress.

The services of a skilled graphic designer are worth the expense.

The real point of graphic design, which comprises both pictures and text, is clear communication. In the Apple Desktop Interface, everything the user sees and manipulates on the screen is graphic. As much as possible, all commands, features, and parameters of an application, and all the user's data, appear as graphic objects on the screen.

Graphics are not merely cosmetic. When they are clear and consistent, they contribute greatly to ease of learning, communication, and understanding. The success of graphic design is measured in terms of the user's satisfaction and success in understanding the interface.

If you design your icons and other graphics on the target screen, rather than on paper, you'll take advantage of whatever that screen has to offer and you'll have the best design possible. Not all screens are alike. For example, a Macintosh Plus has approximately square pixels that are either black or white. Apple II pixels aren't square, and can be any of many different colors.

Visual consistency

The purpose of visual consistency is to construct a *believable environment* for users. Because such concepts as storing documents in folders and throwing things away in the trash can be the same both in the real world and in the Apple Desktop environment, users don't have to relearn them to begin working. This transfer of skills is one of the most important benefits of a consistent interface, especially for beginning users.

Photographic realism isn't essential; the important thing is that the user understands the intended meaning. A well-designed symbol or caricature can convey meaning better than a completely realistic picture.

If images don't efficiently convey meaning, the user is lost in an environment of random objects, and communication breaks down. Graphics—the icons, windows, dialog boxes, and so on—are the basis of effective human-computer dialog and must be designed with that in mind.

Simplicity

Simple design is good design. Don't clutter the screen with too many windows, overload the user with complex icons, or put dozens of buttons in a dialog box. Because icons and dialog boxes must fit in a small space, the messages they convey must be simple and straightforward. Simple designs are easy to learn and to use, and they give the interface a consistent look.

The icons, menus, windows, and other graphic elements on the screen make up a basic language with which the user and computer communicate. The user selects an icon and chooses an action from a menu, effectively telling the computer to "Open MacPaint," for example. For this language to work well, the messages must be simple.

Clarity

Good graphic design begins with an understanding of the situation the user is in or of the problem being solved. A picture isn't always the answer—sometimes words do the job better. Make graphics clear and readable. Try them out on real users, not just on your fellow artists or programmers. The most important part of the graphic should be recognized first, then the second most important part, and so on. Use visual cues such as arrows, movement, and the arrangement of elements to direct the eye to the correct place. The symbols used in different kinds of alerts tell the user if the alert is a note, caution, or warning.

Animation, *when used sparingly*, is one of the best ways to draw the user's attention to a particular place on the screen. For example, users soon learn that the quickest way to find a pointer on a busy screen is to move the mouse, making the pointer move on the screen. Animated pointers reassure the user, during a lengthy process such as saving a large document to disk, that the system is alive and well.

A strategy for programming

The Apple Desktop Interface relies on some distinctive models for programming, some of which are unfamiliar even to experienced programmers.

To help the programmer make use of this interface, and to carry through in these models, some Apple hardware systems provide an abundance of tools in ROM. The developer derives two major advantages from using ROM-based tools and resources: compatibility and efficiency. The more a program bypasses or replaces these tools and resources, the more likely that sooner or later it will be incompatible with new products or features.

Although a developer might know a more direct way of getting information or performing an operation, using system-provided features ensures hardware independence. For example, always reference the proper data structures to determine the current size of a screen rather than using the constant values for current hardware.

The next sections deal with some important programming issues that are at the heart of the Apple Desktop Interface.

Modelessness

With few exceptions, a given action on the user's part should always have the same result, irrespective of past activities.

Modes are contexts in which a user action is interpreted differently than the same action would be interpreted in another context. In other words, the same action, when completed in two different modes, results in two different reactions. A mode typically restricts the operations that the user can perform while the mode is in effect.

Because people don't usually operate modally in real life, dealing with modes in computer environments gives the impression that computers are unnatural and unfriendly.

A mode is especially confusing when the user enters it unintentionally. When this happens, familiar objects and commands may take on unexpected meanings and the user's habitual actions cause unexpected results.

Most conventional software uses modes heavily. It's tempting to use modes because they sometimes make programming easier. But if you yield to the temptation too frequently, users will consider using your application a chore rather than a satisfying experience.

This is not to say that you should never use modes in applications. Sometimes a mode is the best way out of a particular problem. Most of these acceptable modes fall into one of the following categories:

- Long-term modes, such as doing word processing as opposed to graphics editing. In this sense, each application is a mode.
- Short-term "spring-loaded" modes, in which the user must constantly do something to maintain the mode. Examples would be holding down the mouse button to scroll text or holding down the Shift key to extend a text selection.
- Alert modes, in which the user must rectify an unusual situation before proceeding. Keep these modes to a minimum.

Other modes are acceptable if they do one of the following:

- They emulate a familiar real-life situation that is itself modal. For example, choosing different tools in a graphics application resembles the real-life choice of physical drawing tools. MacPaint and other palette-based applications exemplify this use of modes.
- They change only the attributes of something, but not its behavior. The boldface and underline modes of text entry are examples.
- They block most other normal operations of the system to emphasize the modality, as in error conditions incurable through software (for example, a dialog box that disables all menu items except Close).

If an application uses modes, there must be a clear visual indication of the current mode, and the indicator should be near the object most affected by the mode. A good example is the changing pointer in MacPaint: it looks like a pencil, paintbrush, spray can, or eraser, depending on the function ("mode") the user has chosen. It should also be very easy to get into or out of the mode (such as by clicking on a different palette symbol).

No mode should ever prevent a user from saving a document or quitting the application.

The event loop

Applications are prepared for the user to do anything at any time.

The event loop is central to programming for the Apple Desktop Interface. The event loop is the central routine of any application. An application doesn't have to expect a certain set of events in a particular order, but constantly looks for inputs (mouse actions, keystrokes, disk insertions) that can occur in any order and to which it must respond in specific ways.

This approach to programming contrasts with programs that systematically limit the alternatives available to the user, assuring that the user follows the "right" path to the "right" place. Instead, the emphasis is on responding to each local request the user makes, leaving the responsibility for the final destination with the user. In each context, the widest possible range of user activities should be allowed. For example, there's no reason not to let the user set printing options before there's anything to print.

Reversible actions

Always provide a way out.

Because the Apple Desktop Interface encourages users to be active, they often request something they don't really want. To encourage such deliberate (though often unplanned) activities and to give users a sense of control over these activities, programmers should make actions reversible whenever possible. Users should, for example, be able to cancel activities easily, particularly those that are unexpectedly involved. They should also have a range of deliberate choices to confirm that they do want to do something particularly drastic, complex, or time-consuming.

The screen

The screen is the stage for human-computer interactions.

In many computer systems, most of the activity is invisible. Users make inputs, to which the computer returns elaborate responses after some amount of calculation. The screen then becomes the "mail slot" through which exchanges between the user and the computer system take place.

In the Apple Desktop Interface, the screen displays a representation of the "world" that the computer creates for the user. On this screen is played out the full range of human-computer interactions. Initially, it provides the alternatives; then it reflects the results of requested activities; then it again shows the alternatives; and so on. And it does this in an extremely well-defined way. The details are in Chapters 2 and 3.

Though the screen is itself not the interface—the functionality provided by the interface elements is the interface—the screen does play a central role, and managing it is one of the programmer's most important tasks.

Plain language

Communicate with the user in concise and simple terms.

The Apple Desktop Interface is approachable by the unsophisticated user. It requires no special "language." In fact, much of the user-computer interaction is graphic. The user points to objects on the screen and selects from available lists; the computer changes text and graphics at the user's request.

Occasionally, the computer must display textual messages, either to describe a particular situation or to ask the user for a specific decision. In these instances, the phrasing must be very direct and unambiguous. It should inform users directly of the options available.

Whenever words are involved, the design team should include a skilled writer.

User testing

The primary test of the user interface is its success with users.

Can users understand what to do and can they accomplish the task at hand easily and efficiently? The best way to answer these questions is to put them to the users.

The design process

Users should be involved early in the design process so that changes in the basic concept of the product can still be made, if necessary. While there's a natural tendency to wait for a good working prototype before showing the product to anyone, this is too late for the user to have a significant impact on design. In the absence of working code, you can show test subjects alternate designs on paper or storyboards. There are many ways that early concepts can be tested on potential users of a product. Then, as the design progresses, the testing can become more refined and can focus on screen designs and specific features of the interface.

Test subjects

There is no such thing as a "typical user." You should, however, be able to identify some people who are familiar with the task your application supports but are unfamiliar with the specific technology you are using. These "naive experts" make good subjects because they don't have to be taught what the application is for, they are probably already motivated to use it, and they know what they need to accomplish the task.

You don't need to test a lot of people. The best procedure for formative testing (testing during the design process) is to collect data from a few subjects, analyze the results, and apply them as appropriate. Then, identify new questions that arise and questions that still need answers, and begin all over again—it is an iterative process.

Procedures

Planning and carrying out a true experimental test takes time and expert training. But many of the questions you may have about your design do not require such a rigid approach. Furthermore, the computer and application already provide a controlled setting from which objective data can be gathered quite reliably. The major requirements are

- to make objective observations
- to record the data during the user-product interaction

Objective observations include measurements of time, frequencies, error rates, and so forth. The simple and direct recording of what someone does and says while working is also an objective observation, however, and is often very useful to designers. Test subjects can be encouraged to talk as they work, describing what they are doing or trying to do, what they expect to happen, and so on. This record of a person's "thinking aloud" is called a *protocol* by researchers in the fields of cognition and problem-solving, and is a major source of their data.

The process of testing described here involves the application designer and the test subjects in a regular cycle of feedback and revision. Although the test procedures themselves may be informal, user testing of the concepts and features of the interface should be a regular, integral part of the design process.

Designing for disabled people

Computers hold tremendous promise for people with many kinds of disabilities. In terms of increasing productivity and mobility, computers can have a far greater impact on disabled people than on other users. But too often, computers become obstacles rather than enablers, because many disabilities make it hard to use standard computers and software. In most cases, thoughtful hardware design is the solution, but there are things that software designers can do, too.

Many of the modifications that make programs easier for disabled people to use are simple and inexpensive to make, and they often have a welcome and unexpected side effect—the programs are easier for *everyone* to use. Although sidewalk curb cuts are designed to help people who rely on crutches or wheelchairs, they are used and appreciated almost as much by skateboarders and stroller-pushers.

This section describes some of the ways you can design with disabled users in mind. For more information, contact Apple's Office of Special Education Programs.

Vision disabilities

People with vision problems have the most trouble with the output display. The ability of the Macintosh to handle different sizes of text makes it easy to accommodate the needs of many people with vision problems. Software can be designed with a "zoom" feature that automatically increases the size of characters on the screen.

Color is a problem for many people. Don't let people's ability to use your software depend on their ability to distinguish one color from another. Be sure that all information conveyed by color coding is also presented in some other way (by text, position, or highlighting).

Many people have difficulty using the instruction manuals that usually accompany software products, either because they have difficulty reading small print or because they physically can't handle books. These people appreciate having at least the most important part of the manual's text available in electronic form, so that they can display or print it in oversize characters, print it with a Braille printer, or have it read to them through a speech synthesizer. All users benefit from manuals in electronic form, which can quickly be searched for specific topics and keywords.

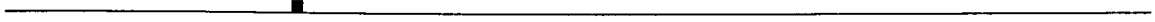
Hearing disabilities

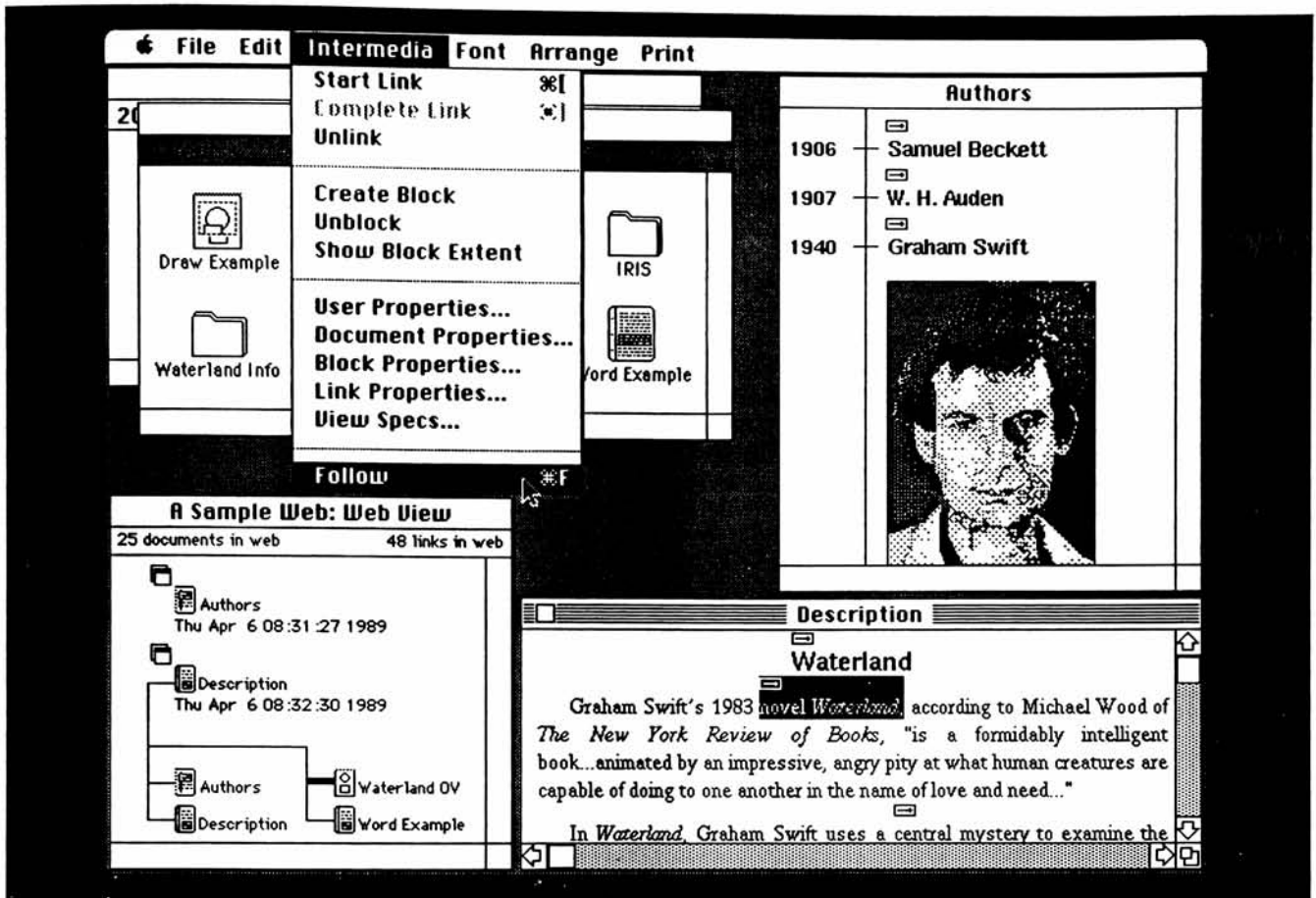
Hearing problems are generally no obstacle to using computers, except when important cues are given only with sound. Aside from the obvious exceptions of music or voice-synthesis applications, software should never rely solely on sound to provide important information. Supplement all audible messages with visual cues, or allow the user to choose visible instead of audible messages.

Other disabilities

People with cognitive or verbal impairments are greatly helped by clear and simple language, icons with obvious meanings, and carefully designed displays. Don't make the user's success depend on his or her ability to *remember* many different things.

Another way to make computers easier for both disabled people and others is to provide macros, making it possible to combine a number of keystrokes and mouse movements into *one* keystroke. The way macros are created and accessed must be clear and simple. It shouldn't be easy for a user to invoke a macro accidentally.





Overview

IRIS Intermedia is an innovative hypermedia development system for the creation and delivery of content-rich, computer-based teaching, training, research, and reference materials. With IRIS Intermedia, users of personal computers in the Apple® Macintosh® II family can establish links between information from many different sources, create branches of thought, blaze trails through complex "webs" of information, and publish shared "ideabases" that continually grow and change.

Built on the A/UX® operating system, IRIS Intermedia provides nonprogrammers with an environment in which to develop sophisticated, interconnected bodies

of information. With IRIS Intermedia, users can create, integrate, and annotate material from different types of media, including text, graphics, timelines, and scanned images. Users can associate ideas intuitively and organize concepts in orderly, yet nonlinear, ways.

Establishing links between related information is as easy as copying and pasting in Macintosh applications. Intermedia webs provide the context in which to collect and navigate these links. Users can browse through information for a general overview of a particular subject, or choose to explore specific ideas in detail.

Developed by the Institute for Research in Information and Scholarship (IRIS) at Brown University, IRIS Intermedia provides the visual framework to explore complex, changing phenomena. It allows users to develop the content they choose, and supports teaching and learning in a broad range of disciplines. Interrelated concepts in multidisciplinary areas, such as English literature or cell biology, can be represented with references to related ideas and events.

With IRIS Intermedia, users can continually reorganize materials and juxtapose information in new and exciting ways.

Features/Benefits (cont.)

- | | |
|---|---|
| <ul style="list-style-type: none">▶ Multiple-window display | <ul style="list-style-type: none">▶ Allows users to view related material simultaneously, helping them to compare and contrast information, as well as to visualize connections. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Consistent user interface | <ul style="list-style-type: none">▶ Makes it easy for users to move rapidly from one application to another without learning new skills.▶ Employs the easy-to-use Macintosh interface, so minimal training time is required for new users. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Infinite Undo and Redo commands | <ul style="list-style-type: none">▶ Gives users complete security, because a previous document state can always be recovered.▶ Encourages users to experiment freely with documents. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Facilities to import existing documents | <ul style="list-style-type: none">▶ Allows users to import documents created in Microsoft Word, MacDraw, MacPaint, and other Macintosh applications. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Built-in dictionary server interface | <ul style="list-style-type: none">▶ Provides networkwide access to a full on-line version of the 125,000-word <i>American Heritage Dictionary</i>. (Dictionary available separately from Brown University.) |

Product Details**Integrated Applications**

IRIS Intermedia combines the ability to create and follow links with a set of powerful, integrated applications. The applications are used for both creating and displaying materials. InterWord, a word processing program with style sheets and formatting, allows users to create texts that are easily read on the screen. InterDraw, a structured graphics editor, can be used in conjunction with InterPix, a scanned-image viewer, to create and display diagrams and bitmapped images. InterVal, a timeline editor, helps manage temporal events by displaying them in chronological order. Like the Macintosh desktop, IRIS Intermedia has a multiple-window display that allows users to work on documents in all of these applications concurrently.

Linking

Linking in IRIS Intermedia is an integrating feature built into the system, so creating and traversing links may be alternated with creating and editing documents. Users can select the "Start Link" command and perform any number of other operations without losing the link reference before selecting the "Complete Link" command. Any part of a document created in IRIS Intermedia can be linked to any part of any other document in the IRIS Intermedia environment. Blocks, which serve as the anchor points for links, can have varying extents, such as a character, a collection of graphics objects, an event on a timeline, or a whole document. A block may be the source or destination for any number of links. Each block has an explainer, which describes the meaning of the block. Link

and block selections are "sticky"; once the endpoints are selected, they will remain the same regardless of editing changes made around them in the document.

The Web View

Links are created and stored not as part of a document, but as part of a separate database called a web. The web is the collection of links in an associated set of documents. A link exists in the context of a particular web, and can only be viewed and followed when both the web and the document are opened. IRIS Intermedia provides a Web View feature that graphically illustrates the connections between linked blocks of information. These views help users to know where they are in a web, to remember what they've already seen, and to find new material to explore.

Features

Benefits

-
- | | |
|---|--|
| <ul style="list-style-type: none">▶ New ways to present information | <ul style="list-style-type: none">▶ Information is organized by association and context, allowing more intuitive access and enabling users to connect seemingly disparate materials in ways that provide insight.▶ Nonlinear presentation encourages critical thinking by providing an efficient means of making connections among materials. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Full set of integrated applications | <ul style="list-style-type: none">▶ Allows users to work with text, graphics, timelines, and bitmapped documents simultaneously.▶ Enables users to create and follow links between any or all of the above media. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Powerful linking capability | <ul style="list-style-type: none">▶ Automates the process of following references.▶ Allows users to be general or specific—users can link related words, paragraphs, diagrams, or even whole documents to provide exact referencing.▶ Highlights the referenced section to provide context. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Multiuser capabilities built on the A/UX operating system | <ul style="list-style-type: none">▶ Allows multiple users to simultaneously access and annotate the same documents. Annotations may be saved as a separate file or shared with other users.▶ Lets users share resources, jointly explore information, and collaborate on projects. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Powerful access rights system | <ul style="list-style-type: none">▶ Permits users to control who has access to documents they create with IRIS Intermedia.▶ Allows system administrators to set different access levels for different groups of users, including read, write, and annotation privileges.▶ Annotation level allows users to create links between documents, without allowing the document contents to be changed. |
| <hr/> | |
| <ul style="list-style-type: none">▶ Seamless environment | <ul style="list-style-type: none">▶ Eliminates the distinction between author and reader that characterizes page-bound text.▶ Encourages browsers to add comments and original material.▶ Aids collaborative efforts by making the annotation process easy.▶ Interactive, self-paced exploration environment inspires browsers to take control of their learning experience. |

IRIS Intermedia

System Requirements

Networked Installation

To use IRIS Intermedia in a networked installation, you'll need:

- ▶ A dedicated Apple Macintosh II, IIx, or IIcx personal computer with a hard disk (minimum storage capacity of 80 megabytes) and at least 4 megabytes of RAM, to act as the network server.
- ▶ A/UX operating system, version 1.1 or later.

- ▶ Additional hard disk space for storing documents and link data.
- ▶ One or more Macintosh II, IIx, or IIcx personal computers, each with at least 4 megabytes of RAM, for client users on the network.
- ▶ One Apple EtherTalk™ NB Card for each Macintosh personal computer on the network, including the network server.

Single-User Installation

To use IRIS Intermedia in a single-user installation, you'll need:

- ▶ An Apple Macintosh II, IIx, or IIcx personal computer with a hard disk (minimum storage capacity of 80 megabytes) and at least 4 megabytes of RAM.
- ▶ A/UX operating system, version 1.1 or later.
- ▶ Additional hard disk space for storing documents and link data is recommended.

Ordering Information

IRIS Intermedia

For U.S. customers, the APDA Order No. is T0260LL/A.

For customers outside the U.S., the APDA Order No. is T0255LL/A.

With your order, you'll receive:

- ▶ Four 800K floppy disks containing the IRIS Intermedia server, client, tutorial, and advanced tutorial
- ▶ *User's Guide*
- ▶ *User's Guide Addendum*
- ▶ *System Administrator's Guide*

- ▶ *Getting Started with IRIS Intermedia*
- ▶ User registration card
- ▶ Support information
- ▶ Information about other Intermedia-related products and literature available from IRIS

IRIS Intermedia is available from:

Apple Programmers and
Developers Association
(APDA™)
Apple Computer, Inc.
20525 Mariani Avenue, M/S 33G
Cupertino, CA 95014
1-800-282-APDA
(1-800-282-2732)
AppleLink®: APDA
Fax: (408) 562-3971

Or:

Institute for Research in
Information and Scholarship
(IRIS)
Brown University
155 George Street, Box 1946
Providence, RI 02912
(401) 863-2001



A Guide for Interactive Interface Design

The discipline of interface design combines elements of the fields of semiotics, cognitive science, computer science and the graphic arts. It is a methodical discipline that also draws on one's intuitive powers and one's ability to think creatively. This guide is my response to being unable to find a practitioner's guide to interactive interface design on the shelves of the library. It's the result of an analysis of the many intricate phases of development of the Graphic Design Archive prototypes 1 and 2. I have focused on design principles for iconographic and metaphorically-oriented desktops like those developed by the Apple Computer, Inc., and my recommendations are meant as general guidelines. Later in this report you will find that I have used this guide as criteria for the criticism of Prototype 1.0 and the description of my thesis project, Prototype 2.0.

1. Examine and Organize the Database

Every database project is governed by preconceived ideas about its ultimate function. Examine the data that you have to work with and make sure that it supports your ideas about the interface. Also be aware that even elegant accessing tools are useless if they allow users to search for non-existent or incomplete data.

a. **Clarify the basic premise** of the project by identifying the reason that it was initiated. Determine whether that reason is still the motivational force behind the project. If there has been a change, determine whether the database can keep up with the changes.

b. **Categorize the data** according to formal similarities. This process may reveal overlapping data structures. Examine the logic of these categories with an open mind. One of the great advantages of creatively organizing a database is that the process can reveal previously hidden relationships between different types of information.

c. **Keep a detailed record** of the organization of the data and any organizational tools that were developed in the process.

2. Develop a user profile

Because different groups of people are attracted to different types of information, the results of step one will help you to develop a general user profile. Targeting your design to this specific user group will guide you in defining the types of accessing tools needed, but limiting the application to this audience might make it obsolete before its time. Build flexibility into your interface design.

a. **Develop a profile** of the individuals or groups who would most likely take a strong interest in the data. Keep in mind that if the extent of the data does not match the expectations of the users, the system will fail to be accepted as a viable interface no matter how well it has been designed.

b. **Make a list** of groups that would definitely not be interested in the data. (Age may be an important determining factor.) Does eliminating these groups allow greater freedom in the design?

c. **Who falls in between?** What is the likelihood that these individuals might take an interest? How can your design provide for these users?

3. Examine the logic for developing accessing methods

It is important to examine traditional methods of accessing data. If the tools for data access allude to and are designed from traditional sources, rather than from computer related ones, the chances of alienating and confusing the user will be greatly diminished.

a. **List the different methods of accessing the database;** for example, list the various organizational categories that could be selected (subject, title, author, et cetera.) Refer to the results of step one while creating this list.

b. **Prioritize the list** from most logical methods of access to least logical. This will direct you in deciding where to begin the process of tool development. Keep in mind that several different information retrieval tools may be necessary to make the database completely accessible. The refinement of these tools should be an ongoing process in the remaining phases of the interface design.

4. Develop a hierarchical flow of operations

Regardless of the number of different accessing tools under development, the final interface will need to be held together by a central accessing methodology, a hierarchical sequence of events necessary for successfully retrieving information from the database.

a. **Review the steps** it will take to retrieve different kinds of information from the database using the planned tools.

b. **Develop a flow chart** that illustrates the retrieval processes.

c. **Examine the chart for redundancies.** Can some of the processes be simplified or combined? Is there a need for more than one way to get to the same basic type of information?

d. **Revise the flow chart** to reflect any new thinking. Revise the accessing tools as needed.

Once the hierarchical order is established, the process of designing the visual interface can begin. The communication of this hierarchical order, through interpretive screen graphics, will help to put the user in control of operations and prevent the frustration that results from premature or unintended interactions. Test your interface ideas as you develop them. Objective input from one or two test subjects can reveal problems that might otherwise be overlooked.

5. Use the concept of physical space in screen design

According to Richard Bolt, author of *The Human Interface*, "People are good at using the space around them for organizing and storing things. They lose this option, though, when they sit down to work with computers. The opportunities and means to use space in dealing with data just aren't there."¹ It is the job of the designer to create an illusion of physical space that allows people to use their organizational skills to keep track of and store their findings. Standard devices useful in this regard include the following:

a. **Metaphors** that allude to an object, a series of objects, a situation, a procedure or a place can communicate a sense of physical space by getting the user to create a mental picture of and identify with the thing to which the allusion is made. The communicative power of the metaphor can also be a drawback. The use of the metaphor can be the cause of unnecessary complexity in screen graphics and unfulfilled or restricted user expectations.

Iconographic and verbal elements of screen design may or may not have metaphoric significance; in either case they can help create an appropriate sense of space, and the following comments apply.

b. **Icons** used as representations of operational procedures are an effective graphic device in interface design. Well designed icons can be informative and save space.

c. **Verbal elements** should be simple, concise and unambiguous.

6. Rules for screen graphics

The Apple Computer Company has published ten fundamental principles of design for their desktop interface.² A copy of these principles can be found in appendix H of this report. The following is a combination of some of their principles with some of my own.

a. The use of a grid system in the design of screen graphics provides the user with a sense of visual stability. For the designer, a grid system provides a consistent template for use in the placement of menus or icons on the computer screen. Designing with a grid system can greatly reduce the amount of time needed for user comprehension and the amount of mouse movement necessary to complete actions.

Construct a basic grid unit from the proportions of a single pixel. This basic unit should, as closely as possible, reflect the size of the typographic unit most commonly used in the interface. Duplicate the unit as many times as necessary to fill the screen. Use the grid to construct a system of margins and axes and to determine the placement of menus, icons and buttons.

There are two books that I have found useful as references for the construction of grids: Grid Systems in Graphic Design by Josef Muller-Brockman,³ and The Grid by Allen Hurlburt.⁴

b. Proportion the size and style of the type, the amount of text and the size of the graphic elements to the overall size of the screen and to the basic unit of the grid. Keep the style of buttons and menus consistent throughout the interface.

c. Determine the hierarchy of importance of textual and graphic information. Each screen should visibly reflect this hierarchy.

d. Recognize the graphic and technical limitations of the software and the computer terminal. The energy spent in trying to get around these limitations might very well be better used in finding and working with the strengths of the available graphics package.

e. Don't get carried away with special effects. Keep your graphics simple, direct, concise and legible. "Good design must communicate, not just dazzle. It must inform, not just impress. Graphics are not merely cosmetic. When they are clear and consistent, they contribute greatly to ease of learning, communication, and understanding. The success of graphic design is measured in terms of the user's satisfaction and success in understanding the interface."⁵

f. Provide consistent visual feedback to user actions, dialog when necessary and ways to back out of or cancel operations. "People expect their physical actions to have physical results, and they want their tools to provide feedback. Users want topics of interest to be highlighted. They want to see what functions are available at any given moment. If grave consequences might follow from any of those functions, they want to know about them - before any damage is done. They want clues that tell them that a particular command is being carried out, or if it cannot be carried out, they want to know why not and what to do instead."⁶