LEARNING WITH HYPERCARD pdf

1: Loper OS Â» Why Hypercard Had to Die

Like the Web, it also allowed for the connections of many different kinds of media. Each card contains a set of interactive objects, including text fields, check boxes, buttons, and similar common graphical user interface GUI elements. Users "browse" the stack by navigating from card to card, using built-in navigation features, a powerful search mechanism, or through user-created scripts. They place GUI objects on the cards using an interactive layout engine based on a simple drag-and-drop interface. This way, a stack of cards with a common layout and functionality can be created. The layout engine is similar in concept to a "form" as used in most rapid application development RAD environments such as Borland Delphi , and Microsoft Visual Basic and Visual Studio. The database features of the HyperCard system are based on the storage of the state of all of the objects on the cards in the physical file representing the stack. The database did not exist as a separate system within the HyperCard stack; no database engine or similar construct exists. Instead, the state of any object in the system was considered to be live and editable at any time. The system operates in a largely stateless fashion, with no need to save during operation. This is in common with many database-oriented systems, although somewhat different from document-based applications. The final key element in HyperCard was the script, a single code-carrying element of every object within the stack. The script was a text field which contents were interpreted in the HyperTalk language detailed below. When the user invokes actions in the GUI, like clicking on a button or typing into a field, these actions are translated into events by the HyperCard runtime. The runtime then examines the script of the object that was the target of the event, like a button, to see if its script object contains code for that event, code termed a handler. If it does, the HyperTalk engine runs the handler, if it does not, the runtime examines other objects in the visual hierarchy. External video "HyperCard Mania! Unlike the majority of RAD or database systems of the era, however, HyperCard combined all of these features, both user-facing and developer-facing, in a single application. This allowed rapid turnaround and immediate prototyping, allowing users to author custom solutions to problems with their own personalized interface. It was this combination of features that also made HyperCard a powerful hypermedia system. Users could build backgrounds to suit the needs of some system, say a rolodex , and use simple HyperTalk commands to provide buttons to move from place to place within the stack, or provide the same navigation system within the data elements of the UI, like text fields. Using these features, it is easy to build linked systems similar to hypertext links on the Web. Objects exist in a message path hierarchy and respond to messages generated by either the user or the system timers for instance. Objects inherit properties and attributes from those above them in the hierarchy. HyperTalk object classes are predetermined by the HyperCard environment, although others can be added by the use of externals see below. HyperTalk is verbose, hence its ease of use and readability. HyperTalk code segments are referred to as "scripts", a term that was considered less daunting to beginning programmers. Each HyperCard object class, contains a set of "properties". For example, buttons are a type of object, and come in standard styles. To determine, say, whether a checkbox style button is in fact checked, a script can simply call the highlight property, [10] which would return either true or false. In a similar way, objects can be analyzed via functions. This is very useful when performing a given action on each separate line of the field. The script that implements the action need only call the function to know exactly the number of lines it must process. Should the field data change, the already coded function call will still be accurate. HyperTalk is a weakly typed language. All variables, and in fact all values of any kind, are stored as typeless character strings handled by the interpreter as numbers or text based purely on context. This has a cost in speed. Variables need not be declared, but rather are created on the fly as they are required. For example, the following expression creates a variable named total, and sets its initial value: Then the expression add 3 to total would result in the string "18" being stored in that variable. Taking this further, a powerful and intuitive structure known as "chunking" allows precise manipulation of text and number strings. It is possible, for example, to have the second character of the value "" the 2 added to the last character of the value "", yielding "". For another example, word 3 of "life is cruel" cruel can be appended after the first word of "Hello world", yielding "Hello cruel world". It would then be possible to put "Goodbye" into the first word of that string, replacing the current value of that word to yield "Goodbye cruel world". The above mentioned terms: HyperTalk supports most standard programming structures such as "if-then" and "repeat". The "if-then" structure is so flexible that it even allows "case" structured code. HyperTalk scripting allows the system to be easily modified and extended. Unlike many procedural languages, and even many scripting languages, HyperTalk proved to be far more accessible to a wide range of users, partly because scripts were more or less readable as English. For instance, put the first word of the third line of field "hello" into field "goodbye" did exactly that. Referring to objects and the items on cards or backgrounds is easy. The example above shows how to access data within a field on a given card, but one can refer to any object in the same fashion, including the stack. All objects can be named or renamed, as in the example above. Adding scripts is also easy. The script may then be edited, saved, and used immediately. Also, HyperCard contains a Message Box, an interactive command-line in a floating window that can execute single lines of script. This also includes the find command, so it doubles as a search dialog. HyperTalk was sufficiently popular that one of its main uses was not as a database, but as a programming tool that empowered ordinary computer users. Thousands of stacks were written and distributed as stackware in the years when HyperCard was widely available. As stated above, programming "for the rest of us", that is, for non-professionals, allowed many thousands of personal applications to be created by individuals with a need for personal software solutions. Some are still in use today. Many hardware and software vendors provided their tutorials as HyperCard stacks, since the application was bundled with all Macs. These were code libraries packaged in a resource fork that integrated into either the system generally or the HyperTalk language specifically; this was an early example of the plug-in concept. Unlike conventional plug-ins, these did not require separate installation before they were available for use; they could be included in a stack, where they were directly available to scripts in that stack. BeeHive Technologies offered a hardware interface that allowed the computer to control external devices. Connected via the Apple Desktop Bus ADB , this instrument could read the state of connected external switches or write digital outputs to a multitude of devices. Externals allow access to the Macintosh Toolbox, which contained many lower level commands and functions not native to HyperTalk, such as control of the serial and ADB ports. It was initially released in August , with the understanding that Atkinson would give HyperCard to Apple only if the company promised to release it for free on all Macs. HyperCard was a huge hit almost instantly. Many people who thought they would never be able to program a computer started using HyperCard for many automation and prototyping tasks, a surprise even to its creator. Project managers found it was being used by a huge number of people, internally and externally. Bug reports and upgrade suggestions continued to flow in, demonstrating it had a wide variety of users. Since it was also free, it was difficult to justify dedicating engineering resources to improvements in the software. It was not lost on Apple or its mainstream developers that the power HyperCard gave to people could cut into the sales of ordinary shrink-wrapped products. This resulted in HyperCard 2. The new version included an on-the-fly compiler that greatly increased performance of computationally intensive code, a new debugger and many improvements to the underlying HyperTalk language. At the same time HyperCard 2. Although stacks HyperCard program documents were not binary-compatible, a translator program another HyperCard stack allowed stacks to be moved from one platform to the other. Then, Apple decided that most of its application software packages, including HyperCard, would be the property of a wholly owned subsidiary called Claris. Many of the HyperCard developers chose to stay at Apple rather than move to Claris, causing the development team to be split. Claris, in the business of selling software for a profit, attempted to create a business model where HyperCard could also generate revenues. At first the freely-distributed versions of HyperCard shipped with authoring disabled. Many users were upset that they had to pay to use software that had traditionally been supplied free and which many considered a basic part of the Mac. Despite the new revenue stream, Claris did little to market HyperCard. Development continued with minor upgrades, and the first failed attempt to create a third generation of HyperCard. During this period, HyperCard began losing market share. Without several important, basic features, HyperCard authors began moving to systems such as SuperCard and Macromedia Authorware. Nonetheless, HyperCard continued to be popular and used for a widening range of applications, from the game The Manhole , an earlier effort by the creators of Myst , to corporate information services, and many thousands in between. In , Apple released the eagerly anticipated upgrade of HyperCard 2. However, these tools were limited and often cumbersome to use because HyperCard still lacked true, internal color support. The resulting HyperCard 3. Development of HyperCard 3. Calhoun and Crow both left Apple shortly after, in In the years that followed, the program saw no more support from Apple, which finally ceased selling HyperCard in March Applications[edit] HyperCard has been used for a range of hypertext and artistic purposes. Before the advent of PowerPoint , HyperCard was often used as a general-purpose presentation program. Examples of HyperCard applications include simple databases, " choose your own adventure "â€”type games, and educational teaching aids. Due to its rapid application design facilities, HyperCard was also often used for prototyping applications and sometimes even for version 1. HyperCard has lower hardware requirements than Macromedia Director.

2: HyperCard - Wikipedia

*Learning with Hypercard [Joseph Hofmeister] on www.enganchecubano.com *FREE* shipping on qualifying offers.*

Click here if you would like to try HyperCard yourself. I was a Hypercard child â€" though our friendship was brief. Our seventh-grade class was led into a room full of brand-new Macintosh Performas. With it, one might persuade a computer to do anything and everything â€" or so it seemed to a child with the attention span to appreciate the wonder. Half a dozen of us were invited back a week later; and then again, and again, for several delicious months. Among these pupils, I was the only one who had already dabbled in programming. And yet there was something magical, something oddly enthralling about Hypercard as a whole. The ease with which a mostly-blank screen could be turned into an interactive, living, breathing graphical toy of my own creation was astounding, exhilarating, and addictive. After the final week, I and one other schoolboy were driven to a distant office building, where we were asked to present our unremarkable creations in front of a darkened lecture hall. The latter was full of somber-faced, suit-wearing adults idly tapping away on costly Apple portables. With their lukewarm applause, the adventure came to its rather boring end. Lacking true English fluency at the time, I never learned exactly who was behind this brief departure from the braindead routine of my early schooling. And without regular access to a Mac given its expense, it may as well have been a Cray as far as my family was concerned I could not return to this fascinating plaything. My development as a programmer continued as it had begun, almost entirely Mac -less and Hypercard-less. Though almost unknown to the sniveling digital trendoids of today, HyperCard was and is one of the most loved software products ever created. It was quite possibly an inspiration for the World Wide Web. Among its satisfied users one could even find the rich and famous. In order to answer this question, it is necessary to actually power up an old Mac or an emulator and try HyperCard on your own skin. So I created one: The materials needed for this recipe were: I fished mine out of a dumpster when I was an undergraduate student. Around half an hour of time. Most of it was spent arranging the screenshots and writing their captions. Create a new HyperCard stack: Now give it a name and save it: Now you have a fresh stack, with a blank card: Re-size the field to reasonable dimensions: Double-click on it and then click on the Script button in the properties dialog: Below is the HyperTalk script that we will attach to the numeric and operator keys of our calculator. It does only one thing: We will give each button the appropriate name: Create the operator keys in the same way: Now, we will need an Equals key. Create a new button: The result will appear: We now have a very simple four-function calculator. Having seen what you just saw, do you now know why Steve Jobs killed HyperCard? Obviously the man is dead and will tell no tales. Does anyone really believe that Mr. He was far too intelligent a man to believe any such thing. One may as well say that you could do everything with a magnetized needle and a steady hand that you could do with a text editor. Or that you could do anything with Roman numerals that you could do with Arabic numerals. Jobs was almost certainly familiar with HyperCard and its capabilities. And he killed it anyway. Apple never again brought to market anything resembling HyperCard. Despite frequent calls to do so. Despite a more-or-less guaranteed and lively market. And I will cautiously predict that it never will again. The reason for this is that HyperCard is an echo of a different world. A world where the personal computer is a mind-amplifier , and not merely an expensive video telephone. What you may not know is that Steve Jobs killed far greater things than HyperCard. He was almost certainly behind the death of SK8. And the Lisp Machine version of the Newton. And we may never learn what else. Jobs had a perfectly logical reason to prune the Apple tree thus. He returned the company to its original vision: A train which goes only where rails have been laid down, like any train, and can travel elsewhere only after rivers of sweat pour forth from armies of laborers. The Apple of today, lacking Steve Jobs â€" probably needs a stake through the heart. Either way, expect no HyperCard or work-alikes from Apple. But how about other vendors? What about open-source projects? Oh, there is no shortage of attempts. And all of them are failures for the same reason: And thus, none of them can readily substitute for it. Sink back into the cube farm hellpit from whence you came. Otherwise, sit down and contemplate the fact that what has been built once could probably be built again. Steve Jobs did not kill the Lisp Machine version of the Newton, but did kill the entire

Newton project. All of the comments pointing out that HyperCard was already obsolete when Steve Jobs killed it are missing the point. Jobs deliberately killed its community by refusing to maintain the product or even release the source. He could have released the source to the welcoming hands of tens of thousands of enthusiasts. All of the modern HyperCard clones all of them substantially more complex and therefore inferior to the original do not add up to a HyperCard community. I am sure that it was not unknown to Jobs. What he truly succeeded in killing was not so much the basic concept of HyperCard but the community. All of the extra features in a more feature-rich system like SuperCard or even VB are not harmless. There is a fundamental difference, especially for a child, between a system which you can fully wrap your mind around and one with countless mystery knobs. Post a comment below or leave a trackback:

3: Upload - HyperCard Online

Specifically, all the students reported that, thanks to their experience with HyperCard, they learned to read more carefully for details and were able to listen better, to speak more, and to write with greater clarity and precision. For instance, one could not depict a dialogue between a patient and a doctor without understanding what was meant in the verbal exchange. In order to use HyperCard, the students also needed to read the instructional sheets over and over again; otherwise, they would not be able to command HyperCard to do what they wanted it to. If a user still failed to understand and follow an instruction, HyperCard made it easier to observe others on the computer screen. Thus, by experimenting on the computer and referring to the instructions repeatedly, the students learned to understand and follow the otherwise difficult instructions. All reported that they understood the last instructional handout much better than the first one. With the specific objective of creating something of their own, reading repeatedly the textbook or the HyperCard instructions was no longer tedious to the students. With the availability of pull-down menus, the students reported that they understood better what was being said. HyperCard apparently made it much easier for the instructor or a fellow student to demonstrate what was meant. It promoted comprehensible input. For example, while the instructor was explaining how to use the paint tools to illustrate a dialogue, she would pull down the Tools menu and actually paint with some of the tools; she would then repeat the instruction again to make sure that everyone understood. As a result of working in pairs, the pairs of students spoke with each other more often, since they needed to exchange ideas frequently between themselves and, when stuck, they needed to ask for help from the instructor or other students. The students were also encouraged to say the words and phrases while typing and to read their written dialogues or texts aloud. This interaction compensated for lack of speaking practice normally associated with working at a computer alone. Another reason for the improvement in their writing might be that typing made them more aware of details in spelling, capitalization, and punctuation, details they might have overlooked before. They were all very willing to correct their mistakes because HyperCard made it easier to do so. They learned from trial and error that they should be exact and careful, especially when typing their commands, or the computer would not obey. For example, whenever there was a spelling error, a missing space, or a misplaced quotation mark, the computer would produce feedback telling the user to check the spelling, punctuation, etc. While they were working, the instructor would move among the students, stopping to assist those needing help. All students reported that their English classroom anxiety level decreased as they became more familiar with the instructor and with their fellow students in the process of using HyperCard. The students were told at the beginning of the course that HyperCard was only to be used as a tool for facilitating the learning of English, and that this was not a computer class where they would be tested on how well they could use the computer. However, they were encouraged to experiment with its use. That assurance from the instructor played an important role in lowering the anxiety level of the students as they were about to embark on a new experience. Decreased anxiety among the students was also attributed to the learner-friendly environment of HyperCard. Likewise, the learner-friendly environment of HyperCard contributed to the consistently high interest levels of the students throughout the course. The students were told to compare HyperCard to a stack of playing cards "stacks" and "cards" are the actual terms used in HyperCard. As with a stack of playing cards, the students could shuffle the cards they created or call up any one card when they wanted to modify anything on that card. But more important than the mere magic of producing and manipulating a stack of playing cards was the learning activities of the students themselves, as they created or selected their own graphics or pictures and wrote their own commands, texts, headings or captions on HyperCard, just as one does when creating a picture book. Only, in this case, the students themselves were both the authors and the illustrators. The project discussed below included as Appendix was created by two of the students and is illustrative of the activities of the students using HyperCard. The first card is the title card, which they titled "Imagination and Creativity. The right and left arrows on their cards can be clicked to navigate among the cards. Real estate agent R: Glad to meet you. What color do you like? My favorite color is green. How much is it? The frequent interactions between partners, with the computer, and with the instructor , which HyperCard required in creating projects of their own, had a strong motivating effect on the students. At the beginning, some students stayed at their computers for long periods of time, reluctantly stepping aside to let their partners get some hands-on experience. However, as they proceeded, each pair of students arrived at a pact which was mutually beneficial. Partners would take turns working on the computers while helping each other with such things as procedural steps and spelling. Throughout the course, the students were continually amazed at what HyperCard could do for them. Once, a student discovered the use of the eraser in the HyperCard software and everyone else went over to watch. Before the students went back to their own computers to experiment with the eraser, the instructor used the occasion to ask them to say the word eraser and then repeat the imperative sentence "Use the eraser to erase your errors" a few times, thus extending and reinforcing their learning experience by having them use words with the "r" sound, a difficult sound in English for many Japanese students. As one can see, the spirit of emulation was quite contagious and inspiring. The students applauded of their own accord after they watched one student display his creation with visual effects to navigate among his cards. Because HyperCard allows its users to do so many things their own ways and in their own styles Stebbins, , the students were highly inspired, enthusiastic, and proud of themselves for the high degree of control they had over their own creative works. The students were so interested in what they were learning and creating that, when the instructor dismissed class, no one would leave the lab until their work was done. At the end of the course, the students proudly presented their projects to the class, electing a champion, a runner-up, and a third-place winner. Both the winners and the others no losers said their use of HyperCard had helped them to learn English. The interactional discourse was marked by use of directives, correction routines, and various forms of repetition employed in the emergent situation as the need to negotiate meaning and exchange ideas arose. Conclusion As an instructional approach to aid beginning ESL students in learning English, the HyperCard project proved to be highly worthwhile, with the results of the experiment supporting the findings in the literature discussed in the first section of this paper. In addition to the enhanced acquisition of English structures, grammar, and vocabulary learned in class, the students acquired the technical vocabulary as well as the procedural knowledge which they could not otherwise have if they had not had the opportunity to experiment with HyperCard. Furthermore, working collaboratively on their own HyperCard projects not only required the students to think on multi-levels in order to link together pieces of work, but also provided a stimulating environment for them to share and discuss in English problems they encountered. As a result, the students became better acquainted with each other, establishing a rapport and a bond with their partners and with the class as a whole. Through the process of learning to use HyperCard to reinforce the language content, the instructor and the students together created a community of eager learners; all students expressed to the instructor a desire to continue to use CALL to helped them learn English. With some computer literacy, teachers should be able to follow the HyperCard manual and create their own well-thought out instructional sheets that combine technical procedures and course contents. A word of caution, however: HyperCard, like any other educational software, such as Hypertext or the Web, is only a tool for facilitating instruction, not a panacea; there should not be any unrealistically high expectations regarding learning gains, since learning occurs best where teachers create well-designed, student-centered, exciting, and supportive learning environments Hyland, Thus, the results of the study can be applied to students of different levels of language proficiency, to other subject areas, and to other learning environments or other cultures. Griffiths discusses the benefits of the adoption of educational software for other languages and cultures as well as descriptions of experiences translating the "Work Room" HyperCard software into Catalan and Bulgarian. With this well-conceived, stimulating tool, teachers may better tap the talents and strengths of all students and inspire creative thinking and active learning. With regard to possible future developments of this line of investigation, I suggest using two randomly selected groups, one with HyperCard and one without, taught by the same teacher, to examine the actual differences in content acquisition. The experiences related to how they learn of those who used HyperCard would then be compared with the experiences of those who did not. The technology age classroom. Computer-assisted language learning as a predictor of success in acquiring English as a Second Language. Computer-assisted cloze exercise in the Adult ESL classroom: Texas Papers in Foreign Language Education, 1 3 , Technology tools in the information age classroom. New York, Oxford UP. What it means and what it costs for small cultures and large cultures. Computers and Education, 22 , A sociolinguistic analysis of low-literate adult learners using educational computer program in the learning of English as a Second Language Doctoral dissertation, Georgetown University, Washington DC, Dissertation Abstracts International, 56, A Using computers with adult ESL literacy learners. National Clearinghouse on Literacy Education. ED Hussin, S. Dissertation Abstracts International, 55, A What can we do to help? System, 21 1 , HyperCard [Computer software, Version 1. Bringing business and liberal arts together via computer-assisted instruction. ED Kim, M. The MacMagic program and its effects on "English as a second language" students: Beryl Buck Institute for Education. ED Marcus, S. Writing Notebook, 7 2 , A study of the dynamics of paired student discourse. System, 21 3 , ED Sampson, D. A technological primrose path? ESL students and computer-assisted writing programs. College English, 53, Process writing for adult ESL and basic education students. ED Stebbins, B. The tool for the classrooms of tomorrow. Computers in the Schools, 7 4 , Her professional interests include CALL, critical L2 education, globalization of English, learning through literature, and multiculturalism.

4: FLOATFX The Ultimate Levitation System with LED Hypercard and DVD | eBay

The HyperCard Listserv is an excellent place to ask questions and get help. However, you should look for the answer in the HyperCard FAQ before posting a question to the listserv. Explanation of the HyperCard listserv and how to subscribe.

Hypercard Amazing Card Illusion A Hypercard illusion is a very puzzling piece of origami style trickery, to sculpt a seemingly impossible card object. Baffling to look at yet easy to make for yourself. Easy to follow video at base of page. This premium illusion has been out for a few years now and Bicycle playing cards showed an article on how to make one on which proved popular. On this page, you will find step by step instructions on how to make one of these fascinating creations. The examples here show the illusion being made with regular playing cards. If you can make them with double backed playing cards, then the illusion will appear even more amazing as there is no way to tell the front from the back. There is also a quick video tutorial at the bottom of this page. The Hypercard as you can see, seems to defy logic and can really baffle your onlookers. The impact is even more impressive as you can make one of these incredible paper sculptures in just a few moments, right in front of your audience. All you will need is a playing card and a pair of scissors. There are three cuts required to be made in the card, the first of which is shown here on the left. Make the cut, a little way into the right side of the card. Cut up as far as the center. Make a second cut over on the left hand side of the card. Again cut up as far as the center. Take care when using scissors, the cutting parts are best left to an adult to do. Now, turn the card over and make a final cut in the middle of the edge of the card. As before cut through to the center of the card. Still holding the card, fold up the bottom right hand portion of the card and align it with the top edge as shown in the photo on the left. Make a little crease along the center fold. Now fold the bottom left hand portion of the pack away from you and push it up till it meets the top edge of the card on the side of the card that is facing away from you. Again make a little crease in the middle portion of the card. This portiion to be creased is shown by the lower thumb in the photo on the left. You can now proceed to untwist your new masterpiece carefully until it resembles something like the photo on the left. You can now place the hypercard on a flat surface. Make sure that you have properly unfolded the card and that it is lying as flat as possible on the surface where it is to be displayed. Your object of wonderment can now be displayed to the amusement of onlookers who, unless they pick up the card and twist around with it, should be really baffled as to how such an impossible object was made.

5: HyperCard: Using Imagination and Creativity to Enhance Learning

From Memex To HyperCard HyperCard was an important tool for eLearning at the birth of the Apple Macintosh computer. But before there was HyperCard, there was Vannevar Bush's Memex.

6: Run Hypercard on Modern Mac OS via Web Browser

In Distant Open Learning system, the lack of socialization between learners appears to be one of the major faults. If the socialization is very important to construct knowledge, it is also.

7: Hypercard Amazing Card Illusion Tutorial

INTRODUCTION For three years, the research-team of the Trigone laboratory of the Lille I University has been working on the setting up of communication systems and tools for distance and Open Learning.

LEARNING WITH HYPERCARD pdf

Little Chicken Chicken Courage on the Causeway (Cover-to-Cover Novels: Adventure) Colin turnbull in his book the mountain people Ekahi (Book One A Holly St. James Romantic Mystery Richard Matheson Harlan Ellison Methods in High Resolution Separation and Analysis of Biological Macromolecules Pages Ohio Revised Code Annual General Index Machine generated contents note: Page The Sylphs of the Season with Other Poems Besides, she reasoned, Leo was safely home in England. She certainly never expected he would return to So Pwn the sat math guide The Labrador missionaries Outline of a practical course in child-rearing Draw and color circulatory system activities for 6th grade We Band of Sisters Pigling and her proud sister. The inventors notebook from the creators notebook series V. 8. Adams, R.L.P. Cell culture for biochemists. V. 1. Historical narrative. Glantz, O. Recent Negro ballots in Philadelphia. A history of Christianity in the world I too had a love story author ravinder singh Foreign silver coins (p. 58-66) The Elusive Embrace Of oliver twist Islamic wazaif ka encyclopedia in urdu Great adventures in the southern Appalachians Network security auditing Cut carbon, grow profits Embodying Inequality The call of the new era In search of a homeland The hiding place on the moor Global Offshore Financial Services Providers Directory Value of Believing in Yourself The electric railway of to-day, by Joseph Wetzler. Exact audio copy guide The principles of ethnological classification Basics of philosophical psychology Gold miners guttersnipes