Self Glorification

Todd Nathan is the Founder and President/CEO of Liveware Corporation. He is a NeRD, NUGhead and basically a totally converted NeXTite. But at times reality hits, and he does work on other platforms to pay the bills. He may be contacted at

Todd Nathan
MegaView Publications
(303) 484-7607
todd@liveware.com

Bugs/Fixes

Here is a non-comprehensive list of some bugs in SS release 1.0, and are already corrected. So if you have purchased 1.0, you can plan on hearing from HSD, or just call them and let them know you need an upgrade to 1.1 when it is available.

- Error recording sound causes you to restart SS.
- Some windows in applications do not come to the front when made key. The best examples of this is the send window in Mail.
- Can't train with Backspace running.

Summary

There are some limitations, like the lack of timing delays in text fields, and sound editing capabilities in Simon Says. I found the sublevels didn't work all that well, and you couldn't have Simon go through the entire sublevel to record it, you had to do each sublevel entry separately. Greg's response was, well you can train the whole application. But there are times when this may not be appropriate, like when I just want to train 3 or 4 commands in a sublevel of 15 or so, but would like to do these 4 or so a once. Lets hope this is fixed in release 1.1, available soon I am told. Aside from this an a couple other short comings, the ease of use was immediately apparent. The minor limitations of this first release could always be worked around. I highly recommend you purchase this software. It is a steal at the price.

System Tested On:	NeXTstation 400/32
Price:	$295.00 MSRP
	$206.50 Educational
Contact::	Dave Peter (dpeter@hsd.com)
	HSD
	(415) 964-1400
	(800) 828-5522
Upgrade Policy:	Shipping Charges for minor upgrades
	Small amounts for major upgrades
Toddster Rating:	4 star in a 5 star system
Release Evaluated:	1.0 (march 16, 1992)

Major Disclaimer

In no way am I being compensated for doing this review. It is for the benefit of the reader, and only the reading audience. If you have any questions, comments or basic complaints about this article's contents, I am the person to contact. Do not bother the HSD people, or Agog people if this article pisses you off. Please, keep it short if you choose to bash me for it's contents. Oh, I don't work for HSD, Agog or NeXT.

paste | jot

Then I just go around highlighting stuff I want to remember and just say 'Jot_Grab'"

The UNIX script execution was only tested lightly as of this writing, but as the previous example shows (thanks Greg) it works.

Manual

The manual is about 70 pages and is well written. I bombed through it in 48 minutes. It is easy reading, and has lots 'o pictures for people like me who hate to read manuals. The fast pace keeps you going. I must point out I had played and figured out every feature except Sublevels prior to reading the manual. So the reading was rather redundant for me, but I can see it is well structured, and written. Dave Peter at HSD said he spent about a week solid on it, and it shows. Good work Dave.

The manual is 10 chapters short, with 3 Appendices and a software license notice. The chapters cover the Introduction, Installation, Getting Stared with Simon Says, Understanding Voice Recognition (not a lecture), Training Voice Commands, Using Voice Control, Adding Applications, Voice-Activated Macros, Advanced Features (sublevels are covered here), and Menu Commands. Appendix A is Troubleshooting, B is Multi-User Installations and C is the software license. You will not find a more brief and complete manual on the market.

I was told the box is nice (by Greg) and the manual is good too. I have neither in hand, so I make no comment on their quality. The online manual is well done, but I hate to print out online manuals. If you are like me, your eyes will be a bit sore when finished with the reading.

Software Support from HSD

I had the opportunity to talk to Dave Peter at HSD, and Greg at Agog about this product. Both where very helpful and understanding of my needs. They both talked rather openly about the market and where Simon fits in. Greg discussed his future a little, without revealing any trade secrets. He did tell me he did Simon Says in a short cycle. Due to the quick development time NeXTstep offers, and Greg's interest in Voice Recognition, we are likely to see him coming out with more advanced and powerful software in the near future. Lets hope so, as this is a good starting point for such a package. Can you say speaker independent, voice independent? I thought you could ;').

The support should be excellent. HSD is not doing much these days except marketing and support. This gives HSD the opportunity to show the industry that a marketing company can work. Greg basically is going back to hacking and HSD is doing the selling, and both will do a great job of it. So don't call Greg, give HSD a call to purchase this software.

some doing, but it could be done.

Sound input worked flawlessly. A short coming was the sound engine only had record, stop and play buttons. I would have liked to edit sounds (my voice) to get rid of the mouse clicking to start and stop the sound recording process. I found the clicks kinda annoying after a while. Dragging sound files into the sound well is supported, so you can quickly add sounds to a macro from your own personal ~/Library/Sounds. The playing of sounds was relatively flawless. On every other playing of sounds, the last 10th of a second was chopped off. I have not heard back from Greg on whether this is a SoundKit bug, or a SimonSays bug.

There is a really cool feature implemented in SimonSays and the sound files which it maintains. Here is a quote from Greg, the author of SimonSays...

> "Todd,
>
> Each application in the browser has a separate file package. The file package contains 2 files. Device independent training information in the voicemodel.smodel and the macros in the commands file. The only thing you will notice if an file is trained is that the voicemodel.smodel file will be larger.
>
> The reason we have this setup is so application developers can distribute a file package for their applicaiton that a user can drop in the SimonSays folder and Simon Says will see it the next time it starts up."

I hope this design encourages developers to include file packages for their applications, thus expediting the training process, and making everyone feel good about using voice recognition as a standard tool in NeXTstep.

I didn't find a need to use the mail response macro module, so I will leave that to the reader to figure out.

Text field entry worked great, and was very useful for things like commonly entered text in a file, or for other redundant tasks in typing. Here is Simon has the most potential. As a suffer of Redundant Motion Syndrome and Carpal Tunnel Syndrome, I would love to reduce the typing volume I perform daily. Such things as headers to letters, signiture and signoffs are basically great candidates for voice commands. What lacks though is a timing mechanism for delays, such as those needed when I login to a remote machine, or telnet, ftp or other network redunancy function. A timing delay is needed for many of these operations, and the text field module doesn't support it in a very robust way. There is a timing delay of 1 second i fyou enter cmd-space in a keyboard macro. But I would like to see specific seconds allowed, and this can be done with timed entries. Greg did send me his 'remember' macro, and it does work... Here it is

> "For instance my favorite macro is called Jot_Grab I have the keyboard object do
>
> Command-c
> Command-space
>
> This sucks highlighted text into pasteboard. The one second delay makes sure the app has time to do the copy. Then I have a unix command to

automagically by Simon, I found I didn't want to sit there for 3-5 minutes training 30 or so commands at 3-5 samples each when all I wanted was a few of the commands trained. Simon should be able to proceed, and ignore without deleting the untrained commands.

Table 1

Program	Number of Commands	Time to Train	AutoLoadCommands
Edit	29 commands	3:45	YES
Terminal	6	4:15	NO

Note: When commands are autoloaded, it saves a huge amount of time. I had to fish around in the menus to find the commands, and then enter them into SimonSays while figuring out a proper command name length so SimonSays would not get confused. The manual states longer commands can sometimes confuse SS.

Also, as can be seen in the Time to Time column, sound lengths can be rather long. This could have made things prohibitive, but the sound data is signal processed into a format that represents the shape of your vocal tract 66 times a second. This is a nice touch.

Macros

Once the training is finished, you can assign macros to the voice command. This is where Simon really shines, for all the former and current Mac crossovers who know QuickKeys and it's successor (booo!!!) QuickKeys2, will love it for this alone. Simon is basically a verbal macro system. You can think of each command as a macro, possibly containing any number of 6 possible types of entries.

There exists, keyboard input, mouse input, sound input, mail response, text field input, and last and not least the UNIX script execution input. All these seemed to work quite well.

Keyboard input is basically useful only for specifying key equivlacies in menus. I tried to enter some lengthy text this way, and most often it was garbled or lost.

The mouse input is mostly for those commands in the menu of an applciation that is not assigned a key-equivalent. Here again, it worked well, but I am pretty sure Greg used the NXJourneler object. I found that I had to preview the menu and find where I was going (even though I knew the menu item existed) and then use the mouse macro facility quickly and execute the menu actions. This is due to the fact that the mouse macro actually responds in the same time frame as when you did the macro recording. I found this to be very annoying, and redundant, for text formatting and other commands that tend to be deeply rooted in the menu structure of a program. What I would like to see is an option to either go with cronologically timing, or quick timing which would ignor all mouse actions that where not 'part' of the actually meant action. I know this might take

Is Simon really Zippy the Pinhead?

Reviewed by Todd Nathan

Installation

Simon Says is a sophisticated, and well thought out speach recognition, and macro event playing application by Agog, Inc., written by Greg Cockroft. Simon Says was sent via the InterNet in a special form. Installation was simple and straight forward, but since I do not have the shipped package with printed manual, I can’t comment on the printed manual. The implementation of a ’key’ to drag into the app license panel was a new and nice touch.

Training

The process of training Simon is simple. The training consists of sampling the user’s voice. You can set the number of samples from 3-6 in the Prefs panel. I found 4-5 being about right. 3 just doesn’t seem to be enough to get a good average voice sample. The more times you sample, you increase the sampling time linearly. It can take quite a while at 5 samples per command to get an application trained. And besides, it gets boring. After the novelty wore off, the people in the office got annoyed by my using SimonSays.

Before you use Simon, you must train it for the basic built in functions of most applications. Some of these are ’Hide_App’, ’Quit_Application’ and so on. Along with generic application commands, you can add an application while in training mode. A nice feature of Simon’s add app button, is it will try to extract (I call it autoload) as many commands from the application’s menus as possible. SimonSays does this by launching the app in question, extracting if possible and then quitting the application. Most of the standard apps where no problem, and then there are a few apps that I found Simon was not able to load commands from. But once the app is added, you can always add additional commands to the list. See Table 1.

The command list is shown in a browser, not unlike how the browser in Workspace Manager works. For a given application, you are at the root level of the browser, and in that ’directory’ there are the commands that belong to that application. Each application has a word in CommonWords to activate it as well as a browser level for the words associated with that application. (SimonSays adds this activation word automatically when you add a new app). SimonSays will not listen for words in an application until all words in the application have been trained. But after loading the commands