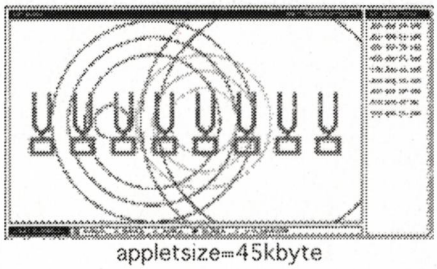
W 406



appletsize=45kbyte

For more infomation nr@hilab.mag.keio.ac.jp

About NetRezonator...

NetRezonator can be defined as a communication system or a chat application exchanging sounds and visuals. Anybody who has terminal computer can conne to this program at the same time. You can enjoy the performance of someone else playing and are free to join the session. When you play, try to make a beautiful harmony together. We'll find ourselves connected strongly by communication beyond words and synchronized each other.

Operating Manual

If you can see sounds and visuals (ripples) after starting the program, it means someone has already been playing in somewhere else. [IP MONITOR] indicates the IP address of player's terminal(client machine). To join the session, chose the tone from the [INSTRUMENT] panel first and click a tuning fork on the [STAGE] area. Eight tuning forks compose an octave (only white keys in piano): C,D,E,F,G,A,B,C2(from the left). This click action is conveyed to all connected terminals (clients) through the server and each program will make changes in sounds and visuals.
If nobody's coming, keep running the program on the back, and wait for a while. Someone should come sooner or later. Your IP addresses are programmed to be written from the third figure for the third party.

System Requirement

Enjoy with a Netscape4.0 or after. Of course Macintosh is available. (low performance) Since this program requires a large CPU power, we recommend you to quit all applications running in backgound before using it.

W 406

Communication Field Beyond Languages

Computer Science in 1990s had brought us various communication tools. E-mail, WWW, BBSs, chat room, the TV conference system, etc.: as we see such technologies, we can no longer imagine our modern life without them --- and these technologies were the keys to create many communities where peop find each other and feel each other as a part of the large, invisible society.
The first concept of Net Rezonator is to 'pursuit non-verbal communication through the Internet.' The significance of non-verbal communication through ou conscious/unconscious movement or expression through multimedia (sounds a images.) But we also should remind ourselves most that computer displays are used for our communication medium in the near future.

Harmony is Strength

Most of our tasks on business are constructed with some kind of a collaboratior Such harmony between people earns a great energy: results of a collaboration are made by organic binds of the members.
The second concept of Net Rezonator is to 'experience the energy created by t harmony.' Costar and to create music and images with strangers. We might finc the great effect of such harmonious, creative work on ourselves, or moreover, on our society.

The Goal of Net Rezonator

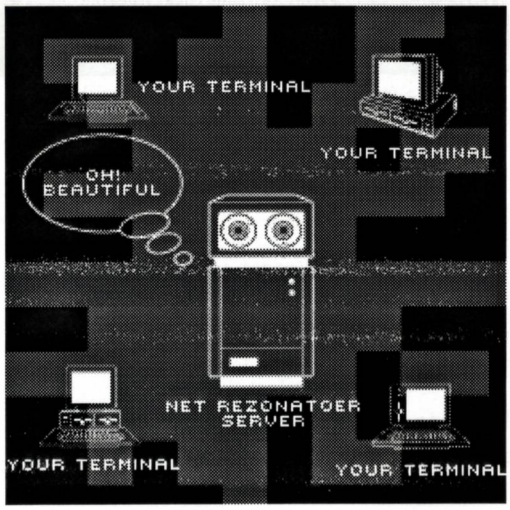
An improvisation found in jazz is a precious space where we can experience the 'synchronicity,' as Young had defined. Musicians communicate with sounds they instantly create, and earns a great ensemble as a session. The third concept ca be defined as such: 'Net Rezonator provides space of improvisation.'
The final goal is to free 'mass unconsciousness of human-being' or 'the directiv recognition', which is restricted by our language and the social system in most of the time. Although we may not reach it, we're sure to find something next inside Net Resonator.

February 8,1998 Koji Ito

References:

The Global Brain by Peter Russell 1982
Digital Harmony by John Whitney 1980
Understanding Media by Marshall McLuhan 1964
Les Cloches De La Terre by Alain Crbin 1994
The Power of Limits by Gyogy Doczi 1981

U 406



How to create a sound on Net Rezonator/ meet sun.audio package

The biggest point of this Applet was how to produce a sound. When creating sounds on Java these days, many people often create each sou as an .au file, and seldom synthesize it while running the program. But this time, since most sounds have a same quality with different scales and also awaring of the time it takes to load, we also programmed synthesizing part. In fact, we weren't really sure what to do in the first place: thanks to search engines, especially Gamelan. And we finally foun the 'sun.audio' package. Usually SUN provides packages by the name of 'java.* ' but as I see from this name, this 'sun.audio' package seems to be more delicate (which means that actions are not 100% guaranteed.) So we started using this without any definite tutorial.

(1)at first, we made a sine wave.
As a first experiment, We synthesized a sine wave. 'Oooh, it ripples!'

(2)changed wave frequencies
I just placed each state of sound waves in an Array, so I thought doublin it will make a sound doubled. And it worked.

(3)changed tunes (or interval)
As I made the second stage, I doubled the frequency and the tune. Well, it's a simple thought, but I was pretty excited when I made it (but the sound wasn't exactly shifting from Do to Re.) Anyway, it was OK for the first place.

(4)resonance
For developing the second step, I placed '* sin(x),' instead of simply making it to double. This made the sound hear waving. This really felt great.

(5)include AU files and try (1) to (4) again
As sine waves successfully came out and went on to the next stage, which is to exchange sound sources with .au files, we suddenly found ou ourselves not having any idea what .au file is. You know, to controll it, you have to know it, of course.

So once again, we started another research and analyzed the AU format In fact, we found nothing... even a single English document didn't come out (or is this a silly question for people who know music?) Well but we found a data in the end, and as we had imagined in the first place, it really was a very simple file format. The file has no data compressed, and there was only a header information with a quite simple wave data (not as easy as a wave data stucked sequentially, though.) And the nam ends up with .au, but the format name is 'mu-Law.' So in the end, we quickly made Filter class, which converts .au data to wave data, and rest of the work went pretty quickly.

(6)a chord
The last stage...the harmony. We felt odd since if we simply add two sounds, the number becomes bigger than the maximize integer, but the sample program also do not have any special method. The result of the harmony seemed to be all right, so we decided that sounds come out independently as we separated threads for different sounds, so we just kept it that way.

February 8, 1998 Kazuo Takagi