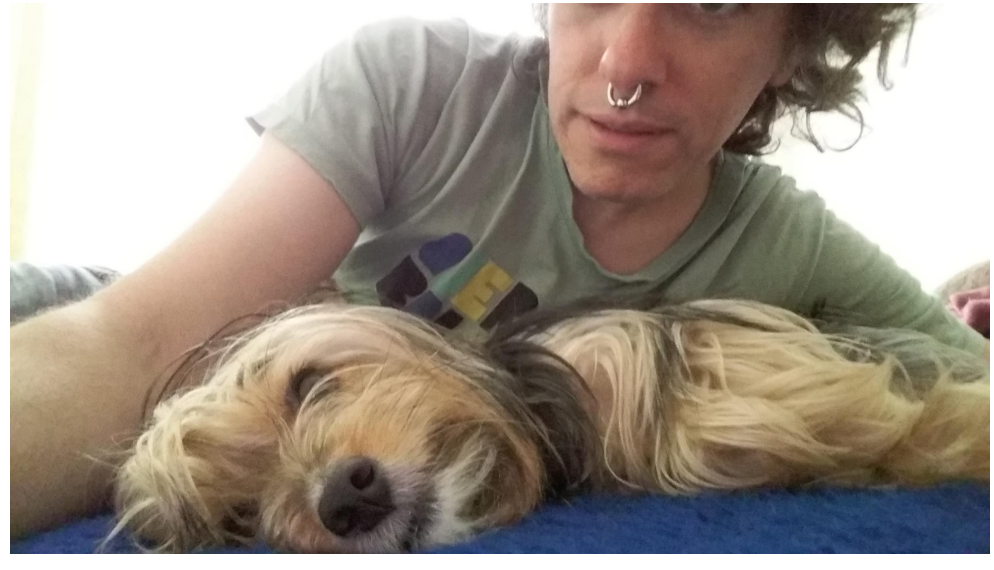
How to record executable notes with eev, and how to play them back

(talk @ EmacsConf 2019; short version)

By:
Eduardo Ochs →
(original author)

Selana Ochs →
(recent contributor)



Prehistory

Eev appeared by accident.

In 1995 I bought a computer that could run GNU/Linux and a Linux CD. I was studying Maths at the university, and I knew a tiny bit of *NIX and Lisp.

I was fascinated by programming and *NIX seemed to be "the right thing".

My social skills were bad. "Everybody uses vi", but...

My memory was bad too, and I typed slowly and with lots of mistakes, so at one point I gave up learning vi, and tried Emacs.

It was **MIND-BLOWING.**

Prehistory (2)

My notes started to have lots of "elisp hyperlinks" to files and directories that I found interesting...

```
(find-file "/usr/share/emacs/19.24/lisp/")
(find-file "/usr/share/emacs/19.24/lisp/files.el")
```

At first I used '`find-file`', then I created a variant called '`find-fline`', that accepted a line number, then I made it accept a string to search for, then I invented "shorter hyperlinks", like '`find-efile`'...

```
(find-fline "/usr/share/emacs/19.24/lisp/files.el")
(find-fline "/usr/share/emacs/19.24/lisp/files.el" 423)
(find-efile "files.el")
(find-efile "files.el" 423)
(find-efile "files.el" "(defun find-file ")
```

Prehistory (3)

I also wrote a very primitive way to send the region to a shell.
'`M-x eev`' (meaning: emacs-execute-verbosely) wrote
the region into the file '`~/.eev/ee.sh`' (actually '`$EE`'),
and if I typed '`ee`' in a shell it would run the commands
in '`$EE`' in verbose mode, i.e., outputting each command
before executing it.
I had this in my '`~/.zshrc`':

```
export EEVTMPDIR=$HOME/.eev
export EE=$EEVTMPDIR/ee.sh
function ee () { set -v; . $EE; set +v; }
```

The '`. $EE`' was a trick to run the commands
"in the current shell environment", so that things like
'`cd`' and '`export`' would work "as expected"...

Prehistory (4)

In a few months I had created several other kinds of elisp hyperlinks that also supported those extra arguments ("pos-spec-lists") that indicated strings to search for...

Some examples:

```
# (find-man "1 gawk" "Built-in Variables")
# (find-man "1 gawk" "I/O Statements")
# (find-man "1 gawk" "String Functions")
# (find-node "(make)Automatic Variables" "$*" "stem")
# (find-node "(make)Recursion")
# (find-node "(make)Special Targets" ".PRECIOUS")
# (find-node "(make)Chained Rules" ".SECONDARY")
# (find-sh "man -d print > /dev/null")
# (find-sh "man -d print > /dev/null |& grep print")
```

Prehistory (5)

These elisp hyperlinks could be put in comments.
They were always the sexp at the end of the line:

```
#!/usr/bin/tclsh
# (find-fline "~/TCL/inc.tcl")
source $env(HOME)/TCL/inc.tcl
eval [lindex $argv 1]
```

I made ‘`M-e`’ work like ‘`C-e C-x C-e`’:

‘`C-x C-e`’ → ‘`eval-last-sexp`’
‘`M-e`’ → ‘`eval-sexp-eol`’

At that time I was using mostly shell, elisp, Awk, Icon, Tcl, Expect, C, Forth, makefiles, TEX/LATEX, and config files, and all these languages and interpreters supported elisp hyperlinks in comments.

Prehistory (6)

I also created variants of '`M-x eev`' that could send regions of text to other external interpreters besides shells, like Tcl, GDB, LaTeX, and...

Prehistory (7)

...and in a few years I had several megabytes of:

1) scripts with elisp hyperlinks in comments,
2) "executable notes" (or "e-scripts"): a free-form mix of elisp hyperlinks, blocks of code to be sent to shells and other interpreters, and comments in English/Portuguese/whatever...

In 1997 I started to have internet at home, and in 1999 I created a home page and uploaded all these scripts and notes to it, because it was

1) good karma,
2) a way to become a person who deserves hints and help.

Prehistory (8)

I was sure that everybody was using Emacs like that.
At one point I sent an e-mail to one of the Emacs MLs.
In the e-mail I used some elisp hyperlinks and other tricks.
I explained how to use everything there, and I apologized:
"I don't know what are the standard functions that do that,
so I wrote my own ones. Sorry for the ugly names..."

RMS himself answered. It was sort of:
"This looks interesting, and AFAIK no one else is using
Emacs like this. Someone should clean up the code and
document it so that we can included it in Emacs.
Can you do that?"

Prehistory (9)

I started to work to make Eev an official package, and I submitted some code. RMS had some objections...

In 2000 (?) he gave a talk in Campinas, that is 500Km away from where I live (Rio de Janeiro). I went there, and chatted with him after the talk.

He said that users should not be forced to see Lisp.

I was **INCREDIBLY OFFENDED.**

RMS was so startled by my reaction that he walked away.

For me the ability to write elisp one-liners was **THE** thing that completely dissolved the barrier between "users" and "programmers". It was a beautiful catalyst, and the distilled essence of Free Software.

Eev as an ELPA package

2000: "users should not be forced to see Lisp"
'`M-x eev`' was hard to set up
(it needed changing rcfiles)

2019: '`M-x eev`' is now obsolete (→ `eepitch`)
setup is now trivial
lots of sandboxed tutorials: `(find-*-intro)`

I've been using eev to teach Emacs and
Free Software principles to people who
didn't know programming or *NIX

people can learn it by memorizing just three keys:
'`M-e`' ("execute"), '`M-j`' ("jump"),
'`M-k`' ("kill this buffer")

Eev as an ELPA package (2)

In april 10, 2019 Eev became an ELPA package.
The new eev has lots of tutorials, with names like:

```
0. (find-eev-quick-intro)
1. (find-emacs-keys-intro)
2. (find-eev-intro)
3. (find-here-links-intro)
4. (find-refining-intro)
5. (find-pdf-like-intro)
6. (find-eepitch-intro)
7. (find-audiovideo-intro)
8. (find-rcirc-intro)
9. (find-eev-install-intro)
```

The "beginner setup" turns `eev-mode` on and opens `(find-eev-quick-intro)`.

‘M-j’
If you type just ‘`M-j`’ without a numeric prefix
you get a page whose header is this:

```
;; Generated by: (find-eejumps)
;; See: (find-eev-quick-intro "7.1. `eejump'" "`M-j'")
;;      (find-emacs-keys-intro "1. Basic keys (eev)")
;;      (find-emacs-keys-intro "2. Key sequences")
;; For example,
;;     M-1 M-j  runs:  (find-fline "~/TODO")
;;     M-2 M-j  runs:  (find-emacs-keys-intro)
;;     M-5 M-j  runs:  (find-eev-quick-intro)
;; Current eejump targets:
```

The tutorial ‘`(find-emacs-keys-intro)`’ starts with a section about the keys of eev and then has several sections about the main keys of Emacs, with hyperlinks to the Emacs manuals.

A demo

So: '`M-x eev`' is obsolete,
It has been replaced by '`<f8>`', that
always operates on the current line, and moves down.

On normal lines it sends the current line to the
target buffer (usually "`*shell*`")

On red star lines it executes the current line
as Lisp (we use this to set up the target buffer).

The following demo shows how I:
downloaded the Debian source package of `xpdf`,
unpacked it, found the code that implemented
"touchpad scroll", commented it out, and compiled
and installed the changed version.

Links

See:

http://angg.twu.net/#eev

http://angg.twu.net/emacsconf2019.html

http://angg.twu.net/e/emacsconf2019.e.html#short

Thank you! =)