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

(talk @ EmacsConf 2019; long version)

By:
Eduardo Ochs →
(original author)

Selana Ochs →
(recent contributor)

This presentation complements the short one

I prepared two videos for the EmacsConf 2019:
a short one, that should have been < 10 minutes,
but ended up being almost 20 minutes long =(,
and this longer one, about more advanced topics...

The next page shows the abstract of my talk(s)
with the topics of the short video in green
and the topics of the longer video in red.

The themes of the short video and the long video

The abstract that I submitted was:

In the old times we would take notes about a task – think on fieldnotes – by using paper notebooks. Nothing was automatic then: we had to decide what to write and how to write it, we had to switch between "task" and "notes", we had to learn how to write both readably and concisely, and we had to learn how to switch between task and notes without losing focus. Learning these things usually took years.

(continues in the next page...)

The themes of the short video and the long video (2)

In this talk I will present a package called "Eev" that lets us do a modern version of this. Some of its lowel-level modules implement support for many kinds of elisp hyperlinks and a variant of 'isend-mode' called 'eepitch' that lets us select an interpreter and send lines of the current buffer to it; on top of that it has tools that let us create and modify elisp hyperlinks with very few keystrokes.

I will show 1) how to use the elisp hyperlinks and eepitch blocks in already existing notes, 2) how to create elisp hyperlinks and eepitch blocks with very few keystrokes, 3) how to use this to do "task"+"notes" with just a few more keystrokes than we would use to do just "task", 4) how I've been using this to teach Emacs to total beginners.

Some design decisions

Eev appeared (by accident) when I started using GNU/Linux and Emacs in 1995...

Before that I had used MS-DOS for years, and MS-Windows 3.1 a bit.

MS-DOS was bad, GNU/Linux was infitely better than it.
MS-Windows 3.1 was infinitely worse than MS-DOS.
MS-Windows was "user-friendly" in a totally flawed way.

When I started to create my own extensions for Emacs I followed some design principles that were the opposite

(↑ Inspired by **Forth**! Long story!...)

of the ones in MS-Windows 3.1.

I wanted something that felt like a machine whose lid is open.

Digression: music

Selana's name is a homage to a Greek psychedelic band, called Kristi Stassinopoulou. (← I'm a great fan of them)

...so: I listened to lots of Greek music, then some Byzantine music, some arabic music...

...weird scales → weird tunings.

People have different reactions when they listen to arabic music for the first time: some people say "WOW", some other people say "Yuck! This is OUT OF TUNE!"

Yuck vs. Wow

It's the same thing with computer interfaces.
For some people eev is "Wow!", for other people it is "Yuck!"
I'm trying to find more "Wow people".

Eev has a "sexp-based interface" (based on plain text!)...

Yuck vs. Wow (2)

Some people react to music in strange tunings with “Wow”, some people react with “Yuck, this is out of tune”...

The people that say that sexp-based interfaces are “Yucky” usually justify this by saying that sexp-based interfaces are:

1. ugly (no icons, no colors)
2. too technical / scary
   (the code and details “should be hidden”)
3. not user-friendly / weird / wrong
   (different from what they see everywhere)

Motivation: ‘C-h f’ is hacker-unfriendly

If we type ‘C-h f open-line’ we get a ‘*Help*’ buffer with this, where the red parts are hyperlinks...

```
open-line is an interactive compiled Lisp function in `simple.el'.

It is bound to C-o, <insertline>.

(open-line N)

  Probably introduced at or before Emacs version 19.29.

Insert a newline and leave point before it.
If there is a fill prefix and/or a `left-margin', insert them on
the new line if the line would have been blank.
With arg N, insert N newlines.
```

Motivation: ‘C-h f’ is hacker-unfriendly (2)

‘C-h f’ runs ‘describe-function’,
that is a biiiiiiiiiiiiiiiiiiiiiiiiiig function...

I found ‘describe-function’ difficult to understand
and to modify, and even worse, it took me YEARS
(because I am a bad programmer with bad social skills)
to find ways to recreate its hyperlinks to
‘simple.el’, ‘19.29’, and ‘left-margin’
into my own notes...

Eev's variant of ‘`C-h f`’

‘`C-h f`’	→	‘`describe-function`’	(Emacs)
‘`M-h M-f`’	→	‘`find-efunction-links`’	(Eev)

‘`M-h M-f open-line`’ creates a read-write buffer called ‘`*Elisp hyperlinks*`’ containing this...

```
# (find-efunction-links 'open-line)
# (eek "M-h M-f open-line")
# (find-eev-quick-intro "4.2. `find-ekey-links' and friends")

# (find-efunctiondescr 'open-line)
# (find-efunction 'open-line)
# (find-efunctionpp 'open-line)
# (find-efunctiond 'open-line)

# (Info-goto-emacs-command-node 'open-line)
# (find-enode "Command Index" "* open-line:")
# (find-elnode "Index" "* open-line:")

# (where-is 'open-line)
# (symbol-file 'open-line 'defun)
# (find-fline (symbol-file 'open-line 'defun))
# (find-epp (assoc (symbol-file 'open-line 'defun) load-history))
# (find-estring (mapconcat 'identity (mapcar 'car load-history) "\n"))
# (find-estring (documentation 'open-line))
# (find-estring (documentation 'open-line t))
# (describe-function 'open-line)
```

Playing with ‘M-h M-f’

All the hyperlinks in page created by ‘M-h M-f open-line’ are safe! Follow them with ‘M-e’ to see what they do.
Some of them are too technical, like these...

```
# (find-efunctionpp 'open-line)
# (find-efunctiond 'open-line)
# (symbol-file 'open-line 'defun)
# (find-epp (assoc (symbol-file 'open-line 'defun) load-history))
# (find-estring (mapconcat 'identity (mapcar 'car load-history) "\n"))
```

...but you can put the cursor on the name of
the mysterious functions in them and type
‘M-h M-f RET’ to see what they do...

...and all these links are copy-and-pasteable,
because they are plain text!

Three ways to generate elisp hyperlinks

Eev has three basic ways to generate elisp hyperlinks with few keystrokes:

1. The ‘`eewrap-*`’ functions.
   The docstring for ‘`eev-mode`’ says this:

```
Commands to convert the current line into hyperlinks:
  M-F -- wrap its contents in a `find-fline'
  M-M -- wrap its contents in a `find-man'
  M-S -- wrap its contents in a `find-sh'
  (...)
  (Variants:)
  M-T -- generate an "* (eepitch-{xxx,kill,xxx})" block
  M-D -- wrap its contents in three Debian hyperlinks
  M-R -- make a rm/mkdir/cd triple
  M-J -- make a ‘(defun eejump-N ...)’ from N and a hyperlink
```

Links

See:

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

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

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

Thank you! =)

Yuck people / wow people
In the other slides
Yuck people:
weird, too technical, ugly, wrong