ESS in Emacs Org Mode Buffers

part of the ESS intro series Greg Minshall

March 22, 2021

Org Mode

"If Emacs is the distribution, Org Mode is the entire desktop environment one runs on top of it."
Emacs Org Mode is an outline-structured file format and backing software that allows you to

- markup text via asteriskizing, `equalizing`, `tildeizing`, *slashizing*, ~~plusizing~~, underscorizing, etc.
- export a file, or a subtree of it, as, e.g., an .html or .pdf file
- take notes
- create agendas (items with date elements)
- organize your LIFE!
- do calculations – good support for math, LaTeX, etc.
- table support for storing information (including support for formulae)

Babel – Org Mode support for programming

One can write source code in "source blocks", and then that code can be

- evaluated to produce results, which, in turn, may be used as input to another source block. [note there are security issues here: you will need to customize the Emacs variable `org-babel-load-languages`, and you will be prompted each time before a code block is evaluated.]
- exported (either the code, its results, or both) into a .html, .pdf, or other format file
- *tangled*, that is, exported into a separate file, that might be used as input to a compilation or some other packaging step, or be a stand alone script (`Rscript`, say).
- edited, either in place in the Org buffer, or "stand alone" in an Org Source buffer, with the possibility of real-time syntax checking/linting, etc.

R source blocks

In particular, one can have a source block with R code

```
#+begin_src R :results value
"here, the last value executed is the result"
#+end_src
```

To evaluate an entire source block in an Org buffer, type `C-c C-c` or `C-c C-v e`.
To edit this source block in an Org Src buffer, type `C-c '`. To close the Org Source buffer and return to the Org buffer, again type `C-c '`.

Org Src buffers for R

Again, `C-c '` to enter and to exit.
An Org Src buffer for R is an ESS mode buffer.

```
"here, the last value executed is the result"
```

Evaluating code is still initiated with `C-c C-c`, but tends to send just the function at the current location to be evaluated (rather than the whole Org Src buffer), and moves the cursor ("point") to the start of the next function.

:results – an entire slide?

:session for R

Normally, successive evaluations of source blocks do not have access to any state instantiated by previous evaluations (other than emitted results).
However, the `:session NAME` header argument causes all evaluation of a block to occur in a single R process (in an Emacs buffer, under ESS).

```
#+begin_src R :session aRbitRaRyname
a <- rnorm(1)
#+end_src
```

This can be convenient, as it allows you to build up, and examine, a series of functions and variables, ease debugging, etc. [1]

[1] In general, depending on the language of the source block you are using, evaluation with or without sessions may exhibit different behavior. (I think I've noticed this with python.)

Other resources (I)

For Emacs:

- SystemCrafters' playlists include various introductory videos.

For Org Mode:

- The Org Mode web site.
- Rainer König has a series of videos (now also available as a course on Udemy).
- The Org Mode community site ("worg") has a section about Babel.

For Org mode with R:

- Erik Riverson many years ago produced a tutorial that, while dated in some places, is still very useful.
- XXX A copy of a longer exposition of this tutorial XXX

Other resources (II)

For everything:

- the Emacs, etc., "info pages" (`C-h i`), which include information on Emacs, Org Mode, etc. (even R, depending on your installation), provide very detailed information.

There are various ways to connect with other Org Mode users listed here. Similar information for ESS is here.

Farewell

Thank you for "attending" this tutorial. I hope it has given you a sense of Org Mode. This has been more of a teaser than an exhaustive introduction, but the resources we listed above should be enough to ease you into using R with Org Mode.