

Test blocks

A new feature of eev

Eduardo Ochs

<http://angg.twu.net/emacsconf2021.html>

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

Contents

- How to use a test block
(with '`<f8>`'s)
- How to insert a test block
(with '`M-x eeit`')
- How '`eeit`' is implemented
- How to add support for a new language
(with '`M-x find-eeit-links`')
- Etc

```
-- This file:
-- http://angg.twu.net/LUA/square-cube.lua.html
-- http://angg.twu.net/LUA/square-cube.lua
```

```
square = function (a) return a*a end
cube   = function (a) return a*a*a end
```

```
--[[
* (eepitch-lua51)
* (eepitch-kill)
* (eepitch-lua51)

dofile "square-cube.lua"

print(square)
print(cube)
print(square(5))
print(cube(5))

--]]
```

← TEST BLOCK

(LUA SEES IT
AS A COMMENT)

EEV MODE
IS ACTIVE



```
-- This file:
-- http://angg.twu.net/LUA/square-cube.lua.html
-- http://angg.twu.net/LUA/square-cube.lua
```

```
square = function (a) return a*a end
cube   = function (a) return a*a*a end
```

```
--[[
```

```
* (eepitch-lua51)
* (eepitch-kill)
* (eepitch-lua51)
```

SETUP

```
dofile "square-cube.lua"
```

```
print(square)
print(cube)
print(square(5))
print(cube(5))
```

```
--]]
```

LINES
SENT
TO THE
REPL



```
Lua 5.1.5 Copyright (C) 1994-2012 Lua.org, PUC-Rio
```

```
>
> dofile "square-cube.lua"
>
> print(square)
function: 0x55638c326800
> print(cube)
function: 0x55638c3328a0
> print(square(5))
25
> print(cube(5))
125
> □
```

TARGET
BUFFER

(RUNNING
A LUA REPL)



```
-- This file:
-- http://angg.twu.net/LUA/square-cube.lua.html
-- http://angg.twu.net/LUA/square-cube.lua
```

```
square = function (a) return a*a end
cube   = function (a) return a*a*a end
```

```
--[[
```

```
* (eepitch-lua51)
* (eepitch-kill)
* (eepitch-lua51)
```

```
dofile "square-cube.lua"
```

```
print(square)
print(cube)
print(square(5))
print(cube(5))
```

```
--]]
```

TYPING <F8>
ON A RED STAR LINE
EXECUTES THE
REST OF THE LINE
AS LISP
AND MOVES
DOWN

RED
STARS ELISP

```
Lua 5.1.5 Copyright (C) 1994-2012 Lua.org, PUC-Rio
```

```
>
> dofile "square-cube.lua"
>
> print(square)
function: 0x55638c326800
> print(cube)
function: 0x55638c3328a0
> print(square(5))
25
> print(cube(5))
125
> []
```

```
-- This file:
-- http://angg.twu.net/LUA/square-cube.lua.html
-- http://angg.twu.net/LUA/square-cube.lua
```

```
square = function (a) return a*a end
cube   = function (a) return a*a*a end
```

```
--[[
* (eepitch-lua51)
* (eepitch-kill)
* (eepitch-lua51)
```

```
dofile "square-cube.lua"
print(square)
print(cube)
print(square(5))
print(cube(5))
--]]
```

TESTS

LOAD
THIS
FILE

TYPING <F8>
ON A NON-
RED STAR
LINE
SENDS THE
LINE TO
THE TARGET
BUFFER AND
MOVES DOWN

Lua 5.1.5 Copyright (C) 1994-2012 Lua.org, PUC-Rio

```
>
> dofile "square-cube.lua"
>
> print(square)
function: 0x55638c326800
> print(cube)
function: 0x55638c3328a0
> print(square(5))
25
> print(cube(5))
125
> []
```

How to create a test block:

Type `'M-x eeit'`,

or `'M-x ee-insert-test'`,

or `'M-x ee-insert-test-block'`.

The result depends on the language —
`'eeit'` uses the major mode to detect it.

For example, if the major mode is

`'lua-mode'` then `'M-x eeit'` runs

`'ee-insert-test-lua-mode'`.

Look:

```

(defalias 'eeit          'ee-insert-test-block)
(defalias 'ee-insert-test 'ee-insert-test-block)

(defun ee-insert-test-block ()
  "Insert a \"test block\" - an eepitch block in a multiline comment.
If the major mode is `foo-mode' then this function calls
`ee-insert-test-foo-mode' if it exists, and yields an error if
`ee-insert-test-foo-mode' is not defined."
  (interactive)
  (if (fboundp (intern (format "ee-insert-test-%s" major-mode)))
      (funcall (intern (format "ee-insert-test-%s" major-mode)))
      (error "ee-insert-test: Unsupported major mode: %s" major-mode)))

(defun ee-insert-test-haskell-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
{-
* (eepitch-ghci)
* (eepitch-kill)
* (eepitch-ghci)
:load %s
-}
" (buffer-name)))))

```



```
(defun ee-insert-test-haskell-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
{-
* (eepitch-ghci)
* (eepitch-kill)
* (eepitch-ghci)
:load %s
-}
" (buffer-name)))))
```

```
(defun ee-insert-test-js-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
/*
* (eepitch-nodejs)
* (eepitch-kill)
* (eepitch-nodejs)
require(\"./%s\")
*/
" (buffer-name)))))
```

```
/* LANGUAGE
 * (eepitch-shell)
 * (eepitch-kill)
 * (eepitch-shell)
gcc -o test-c test-c.c
./test-c
*/
```

MAJOR MODE

```
-:--- test-c.c All L9 (C/*l eev Abbrev)
-- Haskell
```

```
{-
(eepitch-ghci)
* (eepitch-kill)
* (eepitch-ghci)
:load test-haskell.hs
}

```

← SYNTAX FOR MULTI-LINE COMMENTS

```
-:--- test-haskell.hs All L8 (Fundamental eev)
// Javascript
```

```
/*
(eepitch-nodejs)
(eepitch-kill)
(eepitch-nodejs)
require("../test-javascript.js")
*/
```

WHICH REPL TO RUN HOW TO LOAD THIS FILE

↑ FILENAME

```
/* C
# Org mode
#+begin_comment
* (eepitch-shell)
* (eepitch-kill)
* (eepitch-shell)
#+end_comment

```

```
-:--- test-org.org All L7 (Org eev)
# Python
"""
* (eepitch-python)
* (eepitch-kill)
* (eepitch-python)
exec(open("test-python.py").read(), globals())
"""

```

```
-:--- test-python.py All L8 (Python eev ElDoc)
# Shell
: <<'%%%'
* (eepitch-sh)
* (eepitch-kill)
* (eepitch-sh)
. test-shell.sh
%%%
```

```
-:--- test-shell.sh All L8 (Shell-script[sh] eev)
# Tcl
lindex {} {This is a test block:
* (eepitch-tclsh)
* (eepitch-kill)
* (eepitch-tclsh)
source test-tcl.tcl
}} 0 ;# End of the test block

```

```
-:--- test-tcl.tcl Top L8 (Tcl eev)
```

How to add (or change) the support for a language

Type `M-x find-eeit-links`.

If the current mode is `python-mode`,
this will let you (re)write the function
`ee-insert-test-python-mode`.

It will create a temporary buffer like this:

```
;; (find-eeit-links 'python-mode)
;; (find-efunction 'find-eeit-links)
;; (find-eepitch-intro "3. Test blocks")
;; (find-eepitch-intro "3.1. `find-eeit-links'")
;; (find-eev "eev-testblocks.el" "examples")
```

LINKS TO DOCS

```
;; Current definition:
;; (find-efunction 'ee-insert-test-python-mode)
;; (find-efunctionpp 'ee-insert-test-python-mode)
;; (find-epp (ee-defun-sexp-for 'ee-insert-test-python-mode))
```

LINK TO THE SOURCE CODE

```
<ee-insert-test-python-mode>
;;
;;
(defun ee-insert-test-python-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
--[[
* (eepitch-lua51)
* (eepitch-kill)
* (eepitch-lua51)
dofile \"%s\"
--]]
" (buffer-name))))))
;; Test:
;; (ee-insert-test-python-mode)
```

WRONG! NOT FOR PYTHON! FIX THIS!

EDIT THIS AND SAVE IT TO YOUR ~/.emacs



```
;; <examples> (to ".examples")
;; See: (find-eepitch-intro "3.1. `find-eeit-links'")
```

```
(defun ee-insert-test-c-mode ()
  (interactive)
  (let* ((fnamec (buffer-name))
         (fname (replace-regexp-in-string ".c$" "" fnamec)))
    (insert (ee-adjust-red-stars (ee-template0 "\
/*
* (eepitch-shell)
* (eepitch-kill)
* (eepitch-shell)
gcc -o {fname} {fnamec}
.{fname}
*/
")))))
```

```
(defun ee-insert-test-haskell-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
{-
* (eepitch-ghci)
* (eepitch-kill)
* (eepitch-ghci)
:load %s
-}
" (buffer-name))))))
```

```
(defun ee-insert-test-js-mode ()
  (interactive)
  (insert (ee-adjust-red-stars (format "
/*
* (eepitch-nodejs)
*/
")))))
```

For more info, a real-world example, etc, see:

<http://angg.twu.net/emacsconf2021.html>

By the way:

If you hate the red stars
because they are ‘char 15’s
you can use red bullets
instead – ‘char 8226’s.

Run ‘`M-x find-red-star-links`’
for instructions.