r/Common_Lisp • 3 yr. ago
SlowValue

loading an asdf system from current directory

(while using Emacs/Sly) I repeatedly wanted to load an asdf system from current directory (because of testing foreign code and experimenting).

Editing the `source-registry.conf` every time feels silly. So I wonder if there is an standard convenient way of loading a asdf system from an arbitrary current working directory. Maybe someone of you can answer this?

What I have found out so far:
Doing a

```
(push  (sb-posix:getcwd) asdf:*central-registry*)
```

before `load-system` is, while it works, not recommended because asdf docs disapprove the legacy of modifying `asdf:*central-registry*`.

Then I found out about the quicklisp approach:

```
(pushnew (sb-posix:getcwd) quicklisp:*local-project-directories* :test #'equalp)
(ql:quickload "my-test-system")
```

but this, too, is sort of uncomfortable for such a basic functionality (like using stuff from current directory).

Finally I wrote my own Emacs command to do that in a convenient way, feeling silly, because I could not find a standard approach.

```
(defun my-sly-load-system-from-current-directory (directory)
  "load, via quickload a ASDF system from current (or choosen) directory
Deps: Emacs pakage `sly-quicklisp' and a running CL-REPL"
  (interactive (list (read-directory-name
                      "Directory with ASDF definition: "
                      default-directory)))
  ;; find asd file in directory, code partially stolen from  sly-asdf
  (let ((system-file (cl-find-if #'(lambda (file)
                                     (string-equal "asd" (file-name-extension file)))
                                 (directory-files directory))))
    (when system-file
      ;; tell quicklisp about this directory
      (sly-eval
       `(slynk:eval-and-grab-output
         ,(concat "(pushnew \""
                  directory
```

(feel free to "steal" this above source code!)

5 5 Share

Sort by: Best

Add a comment

Sort by: Best

xach • 3y ago

Modifying `asdf:*central-registry*` is fine if it does what you need. The usual way is to do `(push '*default-pathname-defaults* asdf:*central-registry*)`. Entries in the central registry are evaluated each time, so you could push any form that evaluates to the pathname you want.

5 Reply Share ...

SlowValue OP • 3y ago

The ASDF manual sounds a bit like this is (or will be) deprecated.

2 Reply Share ...

xach • 3y ago

That doesn't matter much - it's easy to re-add if it ever is removed.

2 Reply Share ...

mirkov19 • 3y ago

```
(defun asdf::load-local-system (system &rest keys
                                                    &key
                                  (directory *default-pathname-defaults*)
                                &allow-other-keys)
  "Load asdf system from DIRECTORY (default *DEFAULT-PATHNAME-DEFAULTS*)

Pass all other keywords to ASDF:LOAD-SYSTEM

This function is also an external symbol of ASDF package"
  (let ((asdf:*central-registry*
              (push directory asdf:*central-registry*)))
    (apply #'asdf:load-system system keys)))
```

3 Reply Share ...

mirkov19 • 2y ago • Edited 2y ago

is present (for whatever reason), it offers a choice of which to choose or 0 to bail. Feedback always welcome.

```
(defun asdf::load-local-system (&rest keys
                                                    &key
                                  (directory *default-pathname-defaults*)
                                &allow-other-keys)
  "Load asdf system present in DIRECTORY (default *DEFAULT-PATHNAME-DEFAULTS*)

Pass all other keywords to ASDF:LOAD-SYSTEM

This function is also an external symbol of ASDF package

If no system is present, issue an error.
If multiple systems are present, display them and allow user to select system to
or to enter 0 to bail"
  (let* ((asdf:*central-registry* (push directory asdf:*central-registry*))
         (asd-files (remove-if-not (lambda (filetype)
                                     (string= filetype "asd"))
                                   (uiop:directory-files directory)
                                   :key #'pathname-type)))
    (labels ((load-system (system)
               (format t "Found system \"~a\", loading it" system)
               (apply #'asdf:load-system system keys)))
      (case (length asd-files)
        (0 (error "No ASD files in ~a" directory))
        (1  (let ((system (pathname-name (first asd-files))))
              (load-system system)))
        (t (let* ((systems (mapcar #'pathname-name asd-files))
                  (system
                    (progn
                      (loop :for i :from 1
                            :for system :in  systems
                            :do (format t "~a: ~a~%" i system))
                      (format *query-io* "Select system (0 for none): ")
                      (force-output *query-io*)
                      (let ((index (parse-integer (read-line *query-io*))))
                        (if (zerop index)
                            (format t "No file selected~%")
                            (nth index systems))))))
             (load-system system)))))))
```

2 Reply Share ...