Lisp in a Startup:
The Good, The Bad & The Ugly

Vsevolod Dyomkin
Franz Inc.
2018-04-17 @ ELS
A Bit about Me

LISP, THE UNIVERSE AND EVERYTHING

Overview
Repositories 44
Stars 57
Followers 176
Following 14

Pinned repositories

- **cl-redis**
  Redis client for Common Lisp
  ● Common Lisp ★ 123 ▼ 25

- **cl-nlp**
  Common Lisp NLP toolset
  ● Common Lisp ★ 124 ▼ 20

- **rutilis**
  Radical Utilities for Common Lisp
  ● Common Lisp ★ 80 ▼ 14

- **should-test**
  A minimal yet feature-rich Common Lisp test framework
  ● Common Lisp ★ 23 ▼ 2

- **wiki-lang-detect**
  Text language identification using Wikipedia data
  ● Common Lisp ★ 15 ▼ 4

- **crawlik**
  Lisp web crawler and scraper
  ● Common Lisp ★ 8 ▼ 2

https://vseloved.github.io
TRUST ME, I'M AN ENGINEER
Startup Requirements

* Support research & experimentation
* Flexibility

Not:
* Safety
* Ease of reusing existing solutions
* User-friendliness
* Whatever
The Good: The Language

* Dynamism
* Interactivity
* Uniformity
Dynamism

Not just:
* Dynamic typing
* OO with late binding
Dynamism

Redefine

ALL THE THINGS
Dynamism

Not just:
* Dynamic typing
* OO with late binding

What can (should?) also be redefined:
* syntax
* program flow
* type signatures & class hierarchies
* executable code
* namespaces
* dependencies
* you name it
(signal ‘commited)

I’m just outside town, so I should be there in fifteen minutes.

Actually, it’s looking more like six days.

No, wait, thirty seconds.

The author of the Windows file copy dialog visits some friends.
“Loading Multiple Versions of an ASDF System in the Same Image”
https://www.youtube.com/watch?v=VN58mZsHWXk
DSLs

(defmethod crawlik:scrape ((site geo) source)
  (crawlik:match-html
   source
   '>>(div :class "lastcol ...")
     (>> (div :class "portlet_content")
       (ul (li)
         (>> (span :class "counts") ($ gds))
         (>> (span :class "counts") ($ gse))))))
(defmethod match-expr ((head (eql :$)) &rest tail)
  "Match variable: ($ name &optional expr)."
  (when (or (single tail)
          (let ((*tree* *tree*))
            (apply 'match-expr (rest tail))))
    (:= (? *vars* (first tail)) *tree* *tree* nil)
    t))

(defmethod match-expr ((head (eql :>>)) &rest tail)
  "Match by tree depth-first search: (>> tag &rest contents)."
  (or (when (apply 'match-expr tail)
         !!!)
      (when (listp *tree*)
        (let (matched)
          (dolist (*tree* (rest *tree*) matched)
            (when (apply 'match-expr :>> tail)
              (if *match-multiple*
                (:= matched t)
                (return t)))))))
  (void *tree*)))
Interactivity

nano? REAL PROGRAMMERS USE emacs

HEY. REAL PROGRAMMERS USE vim.

WELL, REAL PROGRAMMERS USE ed.

NO, REAL PROGRAMMERS USE cat.

REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.

EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.

THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.

THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.

WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.

NICE. 'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.

OH YEAH! GOOD OL' C-x M-c M-butterfly...

DAMMIT, EMACS.
The “Hardest” Bug

Socket error in "socket": EPROTONOSUPPORT (Protocol not supported)
   [Condition of type SB-BSD-SOCKETS:PROTOCOL-NOT-SUPPORTED-ERROR]

...the problem (analysis by John Fremlin):

The shared-initialize calls getprotobyname which is not thread safe.

There is little need to call this function at runtime anyway as these proto numbers are quite stable, but getprotobyname_r is a safe alternative (on GNU/Linux).

“Fix”:
(defun sb-bsd-sockets:get-protocol-by-name (name)
  (case name
       (:tcp 6)
       (:udp 17)))

https://bugs.launchpad.net/sbcl/+bug/505497

See also: https://tech.grammarly.com/blog/running-lisp-in-production
Uniformity

http://lisp-univ-etc.blogspot.com/2013/04/errors-in-your-platform.html
If I could change just 1 thing...
If I could change just 1 thing...

Eradicate the .-syntax for cons cells!

(defun |...|-reader| (stream char)
  (when (char= #\. (peek-char nil stream))
    (unless (equal '(#\. #\.)
      (list (read-char stream)
        (read-char stream)))
      (error "malformed ..."))
    '|...|))
3 Flavors of Gains

* incremental
* game-changers
* synergy
The Bad: The Environment

JUST FOR THE SAKE OF ARGUMENT, LET'S SAY THAT—
WAIT, FOR THE SAKE OF WHAT?

ARGUMENT.
OK, COOL, THAT'S TOTALLY A GOOD REASON TO SAY SOMETHING THAT'S WRONG. GOTTA HAVE ARGUMENTS.

I'M JUST PLAYING DEVIL'S ADVOCATE. OK. SO YOU SAW AN ARGUMENT WHERE ONE SIDE WAS THE DEVIL, AND YOU WERE LIKE "MAN, THAT GUY COULD USE AN ADVOCATE."

IT'S... WHY ARE YOU BEING SO DIFFICULT? FOR THE SAKE OF ARGUMENT: ARGH! YAY, IT'S WORKING!
The Modern Programmer
Abandonware
bzip2 bug
Implementation Issues
YMMV

* IDEs/editor support
* Documentation/manuals
* What else?
The Ugly: The FAD

Just like half of the world’s spoken tongues, most of the 2,300-plus computer programming languages are either endangered or extinct. As powerhouses C/C++, Visual Basic, Cobol, Java and other modern language codes dominate our systems, hundreds of other languages are running out of life.

An ad hoc collection of geeks-electronic lexigraphers, if you will—never to save, or even document the lingo of classical software. They’re combing the globe’s 9 million developers in search of coders still fluent in these nearly forgotten linguistic fringes. Among the most endangered are Ada, APL, B (the predecessor of C), Lisp, Oberon, Smalltalk, and Simula.

Code-raker Grady Booch, Rational Software’s chief scientist, is working with the Computer History Museum in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can grok the code. Why bother? “They tell us about the state of software practice, the minds of their inventors, and the technical, social, and economic forces that shaped history at the time,” Booch explains. “They’ll provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an utter failure.” Here’s a peek at the strongest branches of programming’s family tree. For a nearly exhaustive check out the Language List at HTTP://WWW.INFORMATIK.ULM.DE/Java/misc/lang_list.html. - Michael Mendeno
Fight FAD?

* Killer App?
  well, we already have Emacs

* Success Story?
  ITA ($1B), Grammarly, etc.
Fight FAD?

* Killer App?
  well, we already have Emacs

* Success Story?
  ITA ($1B), Grammarly, etc.

... good, but not enough –
we need other positive directions
Rebranding? Common Lisp

http://lisp-univ-etc.blogspot.com.es/2013/01/common-lisp-is-just-lisp.html

Lisp-family languages:
A Company-Champion?
A Community Edition?

LispWorks

ACL
Vsevolod
@vseloved

likes Vladimir's sharpness: "1. Be uncompromising in using #Lisp"
http://goo.gl/JYWP

1:34 AM - 2 Oct 2010