common lisp

Furkan Tektas
CSE 341, Gebze Technical University
October, 2015

John McCarthy*

*Lisp's designer

common lisp

Furkan Tektas
CSE 341, Gebze Technical University
October, 2015

lisp introduced

- if/then/else construct
- recursive function calls
- dynamic memory allocation
- garbage collection
- first-class functions
- lexical closures
- interactive programming
- incremental compilation
- dynamic typing

lisp?

LISt Processor

data types

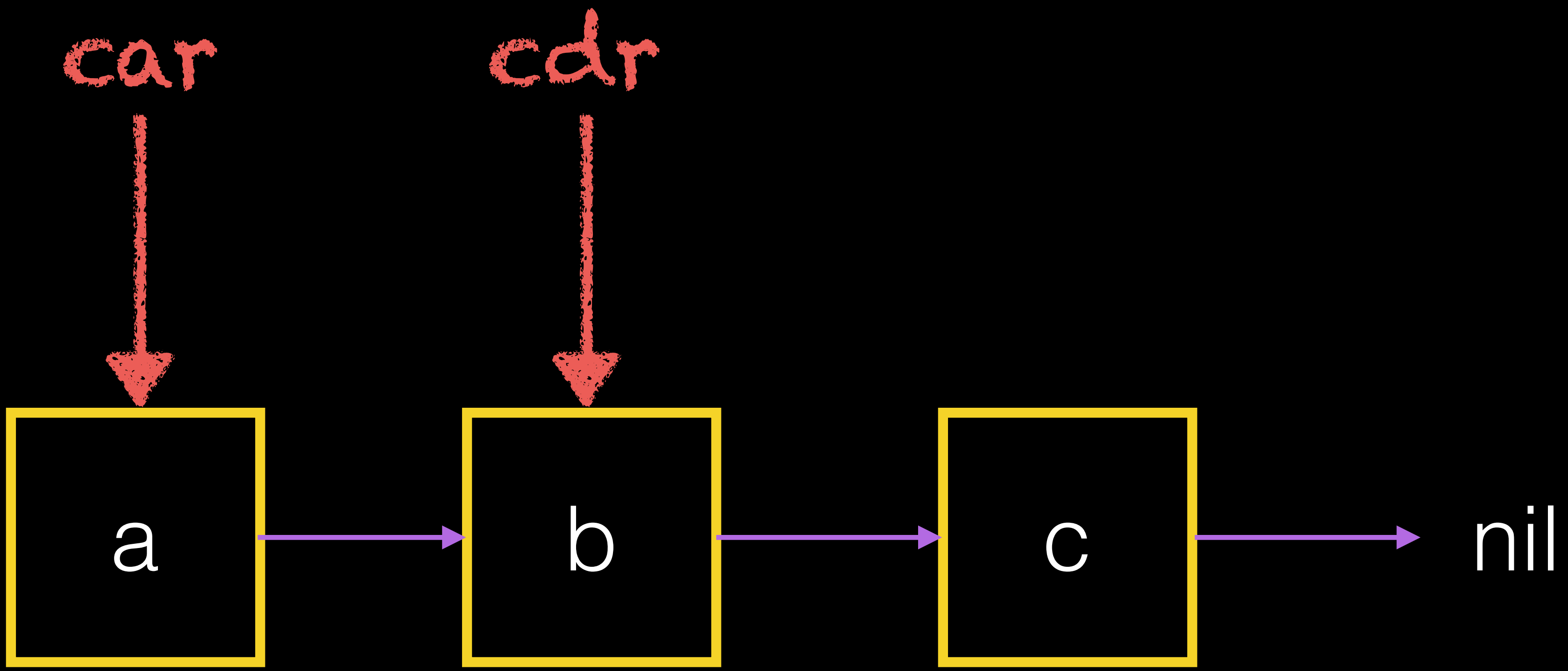
list

atom

lists

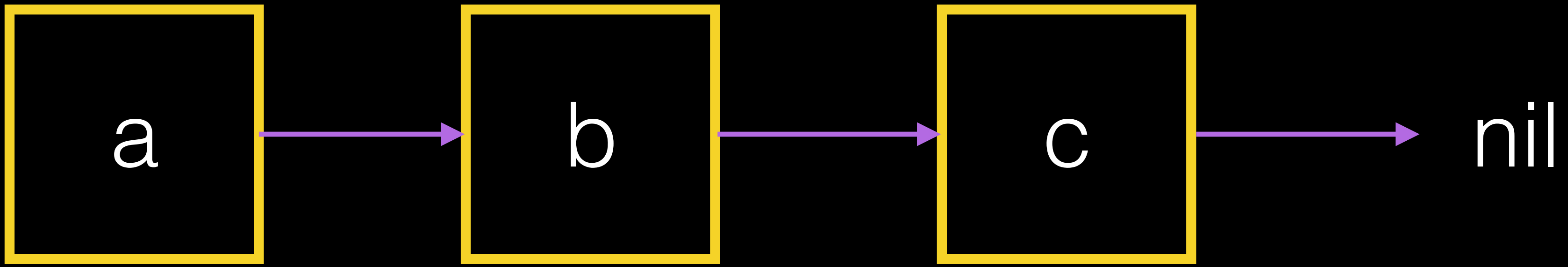
$$\begin{tikzcd}
\boxed{a} \arrow[r] & \boxed{b} \arrow[r] & \boxed{c} \arrow[r] & \text{nil}
\end{tikzcd}$$

lists



list operations

```
'(a b c)
```



```
(car '(a b c))    : a
(cdr '(a b c))    : (b c)
```

nested list

```
'(a b '(c '(d e)))
```

nested list

```
'(a b '(c '(d e)))
```

lots of parentheses

operators

$$\begin{tikzcd}
1+2+3+4 \arrow[r] & (+\ 1\ \ 2\ \ 3\ \ 4) \arrow[d] \\
 & \text{prefix notation}
\end{tikzcd}$$

function definitions

```
(defun topla (a b)
        (+ a b)
)
```

$\longrightarrow$ return value

function definitions

```
(defun topla (a b)
        (+ a b)
)
```

$\longrightarrow$ return value

$\searrow$ not preferred

function calls

```
(defun topla (a b)
      (+ a b)
)
```

```
(topla 3 4)
```

$\longrightarrow 7$

control structures

```
(if t
    1
    2)
```

```
(if nil
    1
    2)
```

control structures

```
(if t
    1
    2)
```

```
(if nil
    1
    2)
```

control structures

```
(if t            (if nil
      1                1
      2)               2)
```

```
   1                 2
```

what can we do with these?

```
(defun factorial (n)
   (if (= n 0) 1
       (* n (factorial (- n 1)))))
```

binding

```
> (let b 2)
  2
> b

*** - SYSTEM::READ-EVAL-PRINT: variable B has no value
The following restarts are available:
USE-VALUE      :R1      Input a value to be used instead of B.
STORE-VALUE    :R2      Input a new value for B.
ABORT          :R3      Abort debug loop
ABORT          :R4      Abort debug loop
ABORT          :R5      Abort debug loop
ABORT          :R6      Abort debug loop
ABORT          :R7      Abort debug loop
ABORT          :R8      Abort main loop
```

```
> (setq a 1)
  1
> a
  1
```

repl

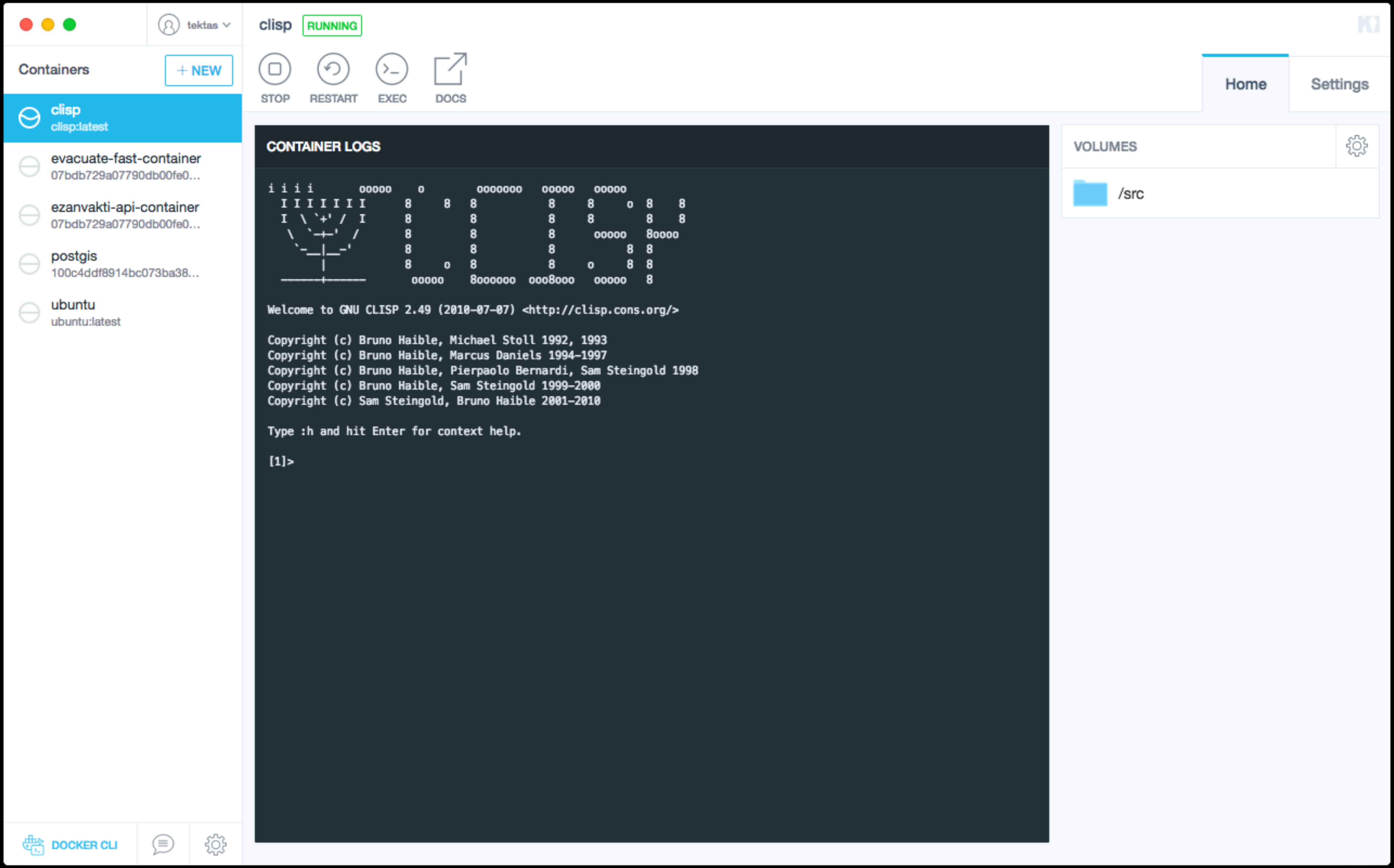
```
# clisp
  i i i i i i i       ooooo    o        ooooooo   ooooo   ooooo
  I I I I I I I      8     8   8           8     8     o  8    8
  I  \ `+' /  I      8         8           8     8        8    8
   \  `-+-'  /       8         8           8      ooooo   8oooo
    `-__|__-'        8         8           8           8  8
        |            8     o   8           8     o     8  8
  ------+------       ooooo    8oooooo  ooo8ooo   ooooo   8

Welcome to GNU CLISP 2.49 (2010-07-07) <http://clisp.cons.org/>

Copyright (c) Bruno Haible, Michael Stoll 1992, 1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2000
Copyright (c) Sam Steingold, Bruno Haible 2001-2010

Type :h and hit Enter for context help.

[1]> (setq a 1)
1
[2]> (* a 5)
5
[3]> (setq a (* a 5))
5
[4]> a
5
[5]>
```

docker

CSE341 clisp container

https://hub.docker.com/r/tektas/clisp/

kitematic



```
i i i i       ooooo    o        ooooooo   ooooo   ooooo
  I I I I I I I      8     8   8           8     8     o  8    8
  I  \ `+' /  I      8         8           8     8        8    8
   \  `-+-'  /       8         8           8      ooooo   8oooo
    `-__|__-'        8         8           8           8  8
        |            8     o   8           8     o     8  8
  ------+------       ooooo    8oooooo  ooo8ooo   ooooo   8

Welcome to GNU CLISP 2.49 (2010-07-07) <http://clisp.cons.org/>

Copyright (c) Bruno Haible, Michael Stoll 1992, 1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2000
Copyright (c) Sam Steingold, Bruno Haible 2001-2010

Type :h and hit Enter for context help.

[1]>
```

references

- Practical Common Lisp: http://www.gigamonkeys.com/book/
- Wikipedia: https://en.wikipedia.org/wiki/Lisp_(programming_language)
- Video Tutorial (Derek Banas): https://www.youtube.com/watch?v=ymSq4wHrqyU