A Tale of Two Lisps

Click to add subtle
20 minute crash course in LISP

LISP is a family of languages.

Though each member of this family has its own quirks, they all share the common attributes of LISP:

- Interactive (REPL)
- Polish Notation (S-expressions)
- LISP Programs are Lists (homoiconic)
Core Concepts of Lisp

Real Metaprogramming
  perl "macros"
  C macros
Code as data
  run-time programming
S-expressions
Macros
  backquote comma syntax
What you expect out of Lisp

Garbage collection
Multi-paradigm programming
  functional
  imperative
  object-oriented
Dynamic typing
Homoiconicity
Lists
  OMG lists
Examples

- Syntax
- Functions
- Macros
Syntax

LISP syntax is very regular.

A LISP program is just a sequence of lists.

Because of this, LISP code can be manipulated by LISP (which is the List processing language after all)
Functions

Value slot vs Function slot
What sorts of things can be functions?
   Things that have been defun'd
   Lambda expressions
   Symbols
Using funcall
Ways you should never bind a function
Macros

Code is data.

Quoting

Eval example

How macros work

  a simple macro example

  a more complex macro example

It is possible to write code at runtime
Two Theories of Computation

- Turing Machine
  - State Machine + Tape

- Lambda Calculus
  - Combinations of Functions
Equal but Different

Turing Machines → procedural, imperative style.

Assembly, ALGOL, FORTRAN, C, etc

Lambda Calculus → compose functions.

Haskell, LISP
Timeline

1956: lisp is conceived
1958: first implementation (IBM 704)
1962-1970ish: Maclisp
   On the PDP-10 and other machines
1981: Zetalisp (Lisp Machine Lisp)
1984: Common Lisp the Language
1985: *LISP / Emacs LISP
1994: ANSI Common Lisp
2007: Clojure 1.0
Early LISP (1960)

Invented by John McCarthy here at MIT. First Implemented by Steve Russel on an IBM 704 computer.
LISP 1

M-Expressions --> S-expressions

collapse[l] = [atom[l] --> cons[l;NIL];null[cdr[l]] -->
[atom[car[l]]] --> l;T --> collapse[car[l]];
T --> append[collapse[car[l]];collapse[cdr[l]]]

DEFINE (((COLLAPSE,
(LAMBDA,(L),
  (COND,((ATOM,L),(CONS,L,NIL))
   ((NULL,(CDR,L)),
     (COND,((ATOM,(CAR,L)),L),(T,(COLLAPSE,(CAR,L))))))
   (T,(APPEND,(COLLAPSE,(CAR,L)),(COLLAPSE,(CAR,L))))))))) ()

(defn collapse* [coll]
  (loop [head [] body coll]
    (let [x (first body)]
      (cond (= (count body) 0) head
            (list? x) (recur head (concat x (next body)))
            true (recur (conj head x) (next body))))))
LISP 1

The first interpreter for LISP on the IBM 704 had:

- Garbage Collection ("reclamation cycle")
- Recursive Functions/Stack ("public push-down list")
- No M-Expressions (they didn't have time to implement them!)
- No Macros
- REPL !! (Flexowriter)
Maclisp

setf
bit vectors
arrays
strings
floats and bignums
no lexical scoping
loop macro
fewer equality forms than CL
flexible read base
Zetalisp

everything in maclisp
complex numbers
rationals
lexical scoping
dynamic closures
object system (flavors)
base 8 (default)
Zetalisp Demo

Dynamic closures
Lists as functions
Common Lisp

Significant influence from Zetalisp

CLOS (common lisp object system)
  - multiple inheritance

Language is static: old libraries still work
*LISP  (Connection Machine)

Common Lisp, with macros which enabled use of parallel variables
EMACS LISP

Mostly used for scripting new functions for editing files with EMACS.

Dynamically Scoped --- interferes with closures.
Clojure


Integrates with Java and the JVM

Advanced support for parallel, thread-safe code.

Immutable data structures.
Lisp Trivia

LISP is second oldest language in modern use.

Used in AutoCAD, Emacs, Reddit, GIMP, Crash Bandicoot, Toy Story.

Gollum from Lord of the Rings was animated on a Symbolics Lisp Machine.
Trivial Trivia

The computer from Jurassic Park was a Connection Machine Model CM-5.