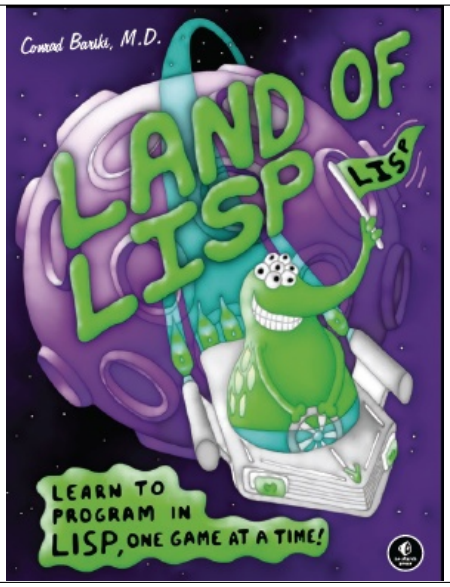


Ch. 16

The Magic of Lisp Macros
Ch. 16

Macros

No other programming language possesses such a simple and comprehensive macro system.

One can even argue that it would be impossible to add this feature to other programming languages, for a simple reason:

The Lisp languages are the only ones in which **computer code** and **program data** are made out of the **same "stuff."**

Not Like C++ Macros

- The `#define` directive in C++ does relatively *simple textual substitution.*
- Lisp macros are much different and more powerful as you will see

Macros Can Simplify Code

```
(defun add (a b)
   (let ((x (+ a b)))        ; let has many ()s
        (format t "The sum is ~a" x)
          x))
```

Define a LET1 macro

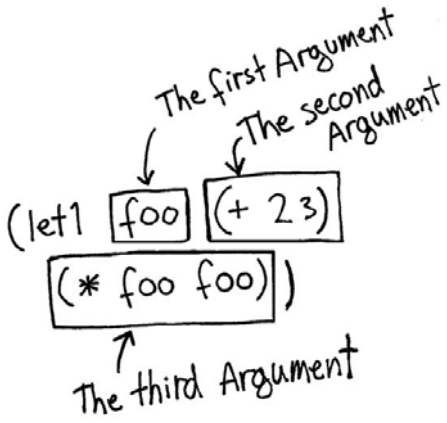
```
(defmacro let1                ; bind one variable (only one)
   (var val &body body)       ; parameter declaration
     `(let                    ; note quasiquote/backquote (`)
        ((,var ,val))         ; comma unquotes backquote!
              ,@body))        ; ,@ unquote and splice into list
```

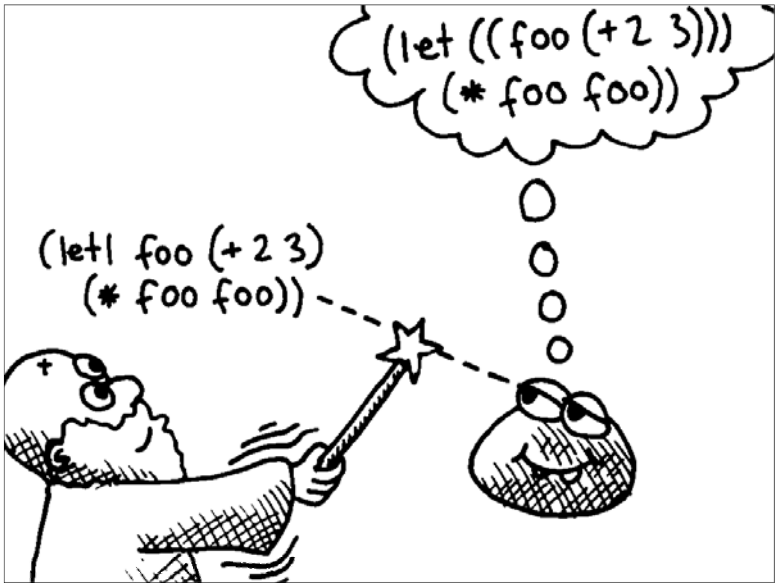
Instead of :

```
> (let ((foo (+ 2 3)))        ;no need for (()) with let1 macro!
              (* foo foo))
25
```

Becomes:

```
> (let1 foo (+ 2 3)
              (* foo foo))
25
```



Macro Expansion

- This magic wand is called macro expansion.
- This is a special transformation that happens to your code **before** the core of the Lisp interpreter/compiler gets to see it.
- The job of the **macro expander** is to
  1. **find** any macros in your code (such as our let1 macro)
  2. **convert** them into regular Lisp code.
- This means a macro is run at a different time than a function is run
- This is called *macro expansion time*.

Summary

- Macros are magic!
- They create and execute new code.
- This enables you to customize your programs
- Macros allow you to write special shortcut macros that can greatly reduce the code necessary to accomplish a task.
- Macros are another means of **control abstraction** - a very powerful one

More Macro Examples

- See the LispMacros slides