

# 函数

aborn

2016-12-04

## Contents

<b>1</b>	<b>什么是函数？</b>	<b>1</b>
<b>2</b>	<b>函数定义</b>	<b>1</b>
2.1	检查一个函数是否定义 . . . . .	1
2.2	函数参数 . . . . .	2
<b>3</b>	<b>函数调用</b>	<b>2</b>
3.1	funcall . . . . .	2
3.2	apply . . . . .	2

---

## 1 什么是函数？

函数是有传入参数的可计算的单元。每个函数的计算结果为函数返回值。大部分计算机语言里，每个函数有其自己函数名。从严格意义来说，lisp 函数是没有名字的。lisp 函数其本质是一个对象，该对象可关联到一个标识符（本书把 Symbol 翻译成标识符），这个标识符就是函数名。

## 2 函数定义

定义一个函数的语法如下：

```
defun name args [doc] [declare] [interactive] body. . .
```

## 2.1 检查一个函数是否定义

检查一个变量是否绑定到函数, `fboundp symbol`, 还有一个函数 (`functionp OBJECT`)

```
(fboundp 'info)                ; t
(fboundp 'setq)                ; t
(fboundp 'xyz)                 ; nil
(functionp (lambda () (message "Anonymous Functions"))) ; t
(fboundp (lambda () (message "Anonymous Functions"))) ; *** Eval error ***
```

## 2.2 函数参数

有些参数是可选的, 当用户没有传时, 设置一个默认值, 下面是一个例子:

```
(defun piece-meal/fun-option-parameter (a &optional b &rest e)
  (when (null b)
    (message "paramete b is not provided")
    (setq b "ddd"))) ;; set to default value
(message "a=%s, b=%s" a b))
```

## 3 函数调用

函数的调用有两种方式, `**funcall**` 和 `**apply**`

### 3.1 funcall

`funcall` 它的语法如下:

```
funcall function &rest arguments
```

因为 `funcall` 本身是一个函数, 因此 `funcall` 在调用前, 它的所有参数都将事先做求值运算。注意参数 **function** 必须为一个 Lisp 函数或者原生函数, 不能为 Special Forms 或者宏 (可以为 lamda 匿名函数)。

### 3.2 apply

`apply` 的语法如下:

```
apply function &rest arguments
```

它与 `funcall` 功能类似, 唯一不同的是它的 `arguments` 是一个列表对象。