

# 函数

aborn

2016-12-04

## Contents

<b>1 什么是函数？</b>	<b>1</b>
<b>2 函数定义</b>	<b>1</b>
<b>3 检查一个函数是否定义</b>	<b>1</b>
<b>4 函数参数</b>	<b>2</b>
<b>5 函数调用</b>	<b>2</b>
5.1 funcall . . . . .	2
5.2 apply . . . . .	2

---

## 1 什么是函数？

函数是有传入参数的可计算的规则。计算的结果为函数返回值。大部分计算机语言里，函数是有一个名字的。从严格意义来说，lisp 函数是没有名字的。注意：函数也是一个 lisp 对象，把这个对象关联到一个 symbol, 这个 symbol 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. . .
```

### 3 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 ***
```

### 4 函数参数

有些参数是可选的, 当用户没有传是, 设置一个默认值, 下面是一个例子:  
“`elisp (defun piece-meal/fun-option-parameter (a &optional b) (when (null b) (message "paramete b is not provided") (setq b "ddd"))) ; set to default value (message "a=%s, b=%s" a b)`”

### 5 函数调用

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

#### 5.1 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)。

#### 5.2 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。