多线程

aborn

2018-05-29

Contents

---

1 多线程

Emacs 从 26.1 版本开始引入了多线程 。它提供了一种简单 (但功能有限) 多线程操作。跟其他编程语言一样，在同一个 Emacs 实例里所有的线程的

内存是共享的。每个线程有其自己运行 Buffer(Current Buffer) 和对应的数据 (Match Data)。注意：下面的文档都是参考 Emacs 的草案手册。

2 基本的线程相关函数

下面介绍线程操作相关的基本函数。

2.1 创建线程

我们可以通过 **make-thread** 函数来创建线程并执行对应的 task。它的语法如下:

```
(make-thread function &optional name)
```

创建一个名为 name 的线程，该线程执行 function 函数，当函数执行结束后，退出该线程。新线程的 Current Buffer 继承当前 Buffer，这个函数返回一个线程对象。可以通过 **(threadp object)** 来判断一个对象是否为线程对象。

2.2 thread-join

thread-join，它阻塞当前执行直到线程执行完成，如果线程已经退出，它立刻返回。

```
(thread-join thread)
```

2.3 thread-yield

执行下一个可执行的线程。

2.4 获取线程名

可以通过 (thread-name thread) 函数来获取线程名。

2.5 线程状态

判断一个线程是否还在执行 (alive)，可以用 (thread-alive-p thread)。

2.6 当前线程

(current-thread) 返回当前线程。

2.7 所有线程列表

获取当前所有正在运行中的线程 (all-threads)。

3 互斥锁 (Mutexes)

互斥 是一种排它锁 (exclusive lock)，在任何时刻，最多只允许一个线程持有互斥锁。也就是说，当一个线程试图获取一个已经被其他线程持有的互斥锁时，它会引发阻塞，直到该互斥锁被释放为止。

3.1 创建一个互斥锁

创建一个互斥锁对象，采用 **make-mutes** 函数，该函数返回一个互斥锁对象，其名字为 name。

```
(make-mutex &optional name)
```

判断一个对象是否为互斥锁使用 (mutexp object)。

3.2 获取/释放互斥锁

```
( mutex-unlock mutex )
```

这个操作会引发阻塞, 直到当前线程获取互斥锁为止。与之相对的有 (mutex-unlock mutex) 释放互斥锁操作。

3.3 with-mutex

```
(with-mutex mutex body)
```

这是一个宏操作，它首先获取一个互斥锁，然后执行 body 里的行为，最后释放互斥锁。

4 条件变量 (Condition Variables)

条件变量 提供线程阻塞直到某个事件发生的机制。线程可以等待一个条件变量，直到别的线程触发这个条件才唤醒。条件变量在某些情况下往往与互斥机制相关联。下面是一个例子：

```
(with-mutex mutex
  (while (not global-variable)
    (condition-wait cond-var)))
```

这里互斥锁保证了原子性。

```
(with-mutex mutex
  (setq global-variable (some-computation))
  (condition-notify cond-var))
```

4.1 创建条件变量

创建条件变量的函数如下：

```
(make-condition-variable mutex &optional name)
```

创建一个与互斥锁 mutex 的条件变量，其名字为 name。判断一个对象是否为条件变量使用 (condition-variable-p object)

4.2 条件等待

```
(condition-wait cond)
```

等待另一个线程去触发条件 **cond**（它是一个条件变量）。这个函数也会阻塞主流程直到条件被触发为止。condition-wait 在等待时会释放与之关联的互斥锁，允许其他线程去获取这个互斥锁从而触发条件变量。

4.3 条件通知

```
(condition-notify cond &optional all)
```

通知 **cond** 条件变量。一般情况下，一个等待线程被 condition-notify 被唤醒，当 all 不是 nil 时，所有等待 cond 的线程都将收到唤醒通知。

4.4 其他函数

1. (condition-name cond) 返回条件变量名
2. (condition-mutex cond) 返回与条件变量相关联的互斥锁