**停止 event\_base**

如果想在移除所有已注册的事件之前停止event\_base循环，可以调用如下函数

int event\_base\_loopexit(struct event\_base \*base, const struct timeval \*tv);

event\_base\_loopexit()：让 event\_base 在给定时间之后停止循环。如果 tv 参数为 NULL, event\_base 会立即停止循环,没有延时

int event\_base\_loopbreak(struct event\_base \*base);

event\_base\_loopbreak ()：让 event\_base 立即退出循环。它与 event\_base\_loopexit (base, NULL)的不同在于, 如果 event\_base 当前正在执行激活事件的回调, 它将在执行完当前正在处理的事件后立即退出

两个函数都在成功时返回 0,失败时返回 -1。

**示例：执行事件循环10秒，然后退出**

void run\_base\_with\_ticks(struct event\_base \*base){

struct timeval ten\_sec;

ten\_sec.tv\_sec = 10;

ten\_sec.tv\_usec = 0;

/\* Now we run the event\_base for a series of 10-second intervals, printing

"Tick" after each. For a much better way to implement a 10-second

timer, see the section below about persistent timer events. \*/

while (1) {

/\* This schedules an exit ten seconds from now. \*/

event\_base\_loopexit(base, &ten\_sec);

event\_base\_dispatch(base);

puts("Tick");

}

}

**判断event\_base释放正常退出**

有时候需要知道对 event\_base 是正常退出的还是因为调用 event\_base\_loopexit()或者 event\_base\_break()而退出的。可以调 用下述函数来确定

int event\_base\_got\_exit(struct event\_base \*base);

int event\_base\_got\_break(struct event\_base \*base);

因为调用 event\_base\_loopexit() 或者 event\_base\_break() 而退出的时候返回 true, 否则返回 false