

# 期中测试 | 这些Redis源码知识，你都掌握了吗？

2021-09-14 蒋德钧

《Redis源码剖析与实战》

[课程介绍 >](#)



讲述：蒋德钧

时长 03:09 大小 2.90M



你好，我是蒋德钧。

时间过得真快，从 7 月 26 日上线到现在，我们已经走过了一个半月的学习之旅，不知道你的收获如何呢？

前面我其实也说过，阅读和学习 Redis 源码确实是一个比较烧脑的任务，需要你多花些时间钻研。而从我的经验来看，阶段性的验证和总结是非常重要的。所以在这里，我特别设置了期中考试周，从 9 月 13 日开始到 9 月 19 日结束，这期间我们会暂停更新正文内容，你可以好好利用这一周的时间，去回顾一下前 20 讲的知识，做一个巩固。

有标准才有追求，有追求才有动力，有动力才有进步。一起来挑战一下吧，开启你的期中考试之旅。

我给你出了一套测试题，包括一套选择题和一套问答题。

- 选择题：满分共 100 分，包含 4 道单选题和 6 道多选题。提交试卷之后，系统会自动评分。
- 问答题：包括 2 道题目，不计入分数，但我希望你能认真回答这些问题，可以把你的答案写在留言区。在 9 月 16 日这一天，我会公布答案。


## 选择题

戳此答题 

## 问答题

### 第一题

Redis 源码中实现的哈希表在 rehash 时，会调用 dictRehash 函数。dictRehash 函数的原型如下，它的参数 n 表示本次 rehash 要搬移 n 个哈希桶（bucket）中的数据。假设 dictRehash 被调用，并且 n 的传入值为 10。但是，在 dictRehash 查找的 10 个 bucket 中，前 5 个 bucket 有数据，而后 5 个 bucket 没有数据，那么，本次调用 dictRehash 是否就只搬移了前 5 个 bucket 中的数据？

 复制代码

```
1 int dictRehash(dict *d, int n)
```


### 第二题

Redis 的事件驱动框架是基于操作系统 IO 多路复用机制进行了封装，以 Linux 的 epoll 机制为例，该机制调用 epoll\_create 函数创建 epoll 实例，再调用 epoll\_ctl 将监听的套接字加入监听列表，最后调用 epoll\_wait 获取就绪的套接字再进行处理。请简述 Redis 事件驱动框架中哪些函数和 epoll 机制各主要函数有对应的调用关系。

好了，这节课就到这里。希望你能抓住期中周的机会，查漏补缺，快速提升 Redis 源码的阅读和学习能力。我们下节课再见！

分享给需要的人，Ta订阅超级会员，你最高得 50 元

Ta单独购买本课程，你将得 20 元

 生成海报并分享

 赞 2  提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 17 | Lazy Free会影响缓存替换吗？

下一篇 期中测试题答案 | 这些问题你都答对了吗？

## 限定福利

限定福利

# 给 Java 工程师 免费送 5 节课

0 元领课 

加赠 PPT

 资深大厂  
专家亲授

 大厂会考  
的面试题

100%

 能落地的  
实战经验

 解决问题的  
思路和方法

## 精选留言 (3)

 写留言



曾轼麟

2021-09-14

问题一：

应该是只搬运了前5个bucket数据，在函数中会初始化empty\_visits为10倍的n，在每次调用改函数的时候最多会遍历10\*n个空元素，并且每次只是递减empty\_visits，最终当empty\_visits为0的时候，方法会直接返回1，结束本次rehash并等待下一次继续，代码如下(返回1代表下次还需要rehash,返回0代表已经完成rehash)：

```
int empty_visits = n*10; /* Max number of empty buckets to visit. */
while(d->ht[0].table[d->rehashidx] == NULL) {
    d->rehashidx++;
    if (--empty_visits == 0) return 1;
}
```

此外注意到后面也有一个while，其主要的目的就是遍历每个bucket底下的链表，代码如下：

```
de = d->ht[0].table[d->rehashidx];
while(de) {
    nextde = de->next;
    .....(此处省略).....
    de = nextde;
}
```

问题二：

以epoll为例子

### 1、epoll\_create

对应的调用函数有aeApiCreate，主要是创建epoll的数组最终整体赋值给aeEventLoop中的apidata，在Redis中所有的IO多路复用是封装成了aeApiState的结构体进行调用的。以epoll为例子，在aeApiState中epfd就是epoll的文件描述符数组。

### 2、epoll\_ctl

对应的函数有aeApiAddEvent和aeApiDelEvent，其中aeApiAddEvent主要是将已经创建的socket文件描述符，通过调用epoll\_ctl方法交给epoll进行管理。而aeApiDelEvent就是移除或者修改对目标socket的管理。

### 3、epoll\_wait

对应的函数有aeApiPoll，调用epoll\_wait后会返回当前已经触发事件(产生了读，写的socket)，并将对应的socket文件描述符指针和读写类型掩码mask，记录在fired数组上等待后续IO线程的处理。

整体来说Redis就是通过封装实现了多个aeApixxx方法，从而抽象了各种IO多路复用的方法，并且能按照操作系统类型选择对应的IO多路复用的方式（在宏定义中修改头文件的方式）。



👍 5



**Milittle**

2021-09-14

1. 第一个问题：empty\_visits=n\*10，空的都跳过，然后打满n个bucket以后，就停止本次rehash，不管empty\_visits满不满无所谓。

2. 从上层到底层：

ae.c:aeCreateEventLoop->ae\_epoll.c:aeApiCreate->epoll\_create

ae.c:aeCreateFileEvent->ae\_epoll.c:aeApiAddEvent->epoll\_ctl

ae.c:aeMain->aeProcessEvents->ae\_epoll.c:aeApiPoll->epoll\_wait



👍 2



**可怜大灰狼**

2021-09-14

1.empty\_visits来控制最大空桶访问数，且是10倍n，所以实际访问桶的数量在[5, 55]。2.在初始化Eventloop的时候会调用aeApiCreate，初始化aeApiState，然后调用epoll\_create打开epoll文件描述符。aeApiAddEvent新增事件和aeApiDelEvent删除事件调用epoll\_ctl来设置epoll\_event。aeProcessEvents获取事件通过aeApiPoll来调用epoll\_wait

