如何基于 Redis 实现延时任务?

> 类似的问题：
>
> - 订单在 10 分钟后未支付就失效，如何用 Redis 实现?
> - 红包 24 小时未被查收自动退还，如何用 Redis 实现?

Redis 是可以用来做延时任务的，基于 Redis 实现延时任务的功能无非就下面两种方案：

1. Redis 过期事件监听
2. Redisson 内置的延时队列

面试的时候，你可以先说自己考虑了这两种方案，但最后发现 Redis 过期事件监听这种方案存在很多问题，因此你最终选择了 Redisson 内置的 DelayedQueue 这种方案。

这个时候面试官可能会追问你一些相关的问题，我们后面会提到，提前准备就好了。

另外，除了下面介绍到的这些问题之外，Redis 相关的常见问题建议你都复习一遍，不排除面试官会顺带问你一些 Redis 的其他问题。

Redis 过期事件监听实现延时任务功能的原理?

Redis 2.0 引入了发布订阅 (pub/sub) 功能。在 pub/sub 中，引入了一个叫做 **channel（频道）** 的概念，有点类似于消息队列中的 **topic（主题）**。

pub/sub 涉及发布者（publisher）和订阅者（subscriber，也叫消费者）两个角色：

- 发布者通过 `PUBLISH` 投递消息给指定 channel。

- 订阅者通过 `SUBSCRIBE` 订阅它关心的 channel。并且，订阅者可以订阅一个或者多个 channel。

在 pub/sub 模式下，生产者需要指定消息发送到哪个 channel 中，而消费者则订阅对应的 channel 以获取消息。

Redis 中有很多默认的 channel，这些 channel 是由 Redis 本身向它们发送消息的，而不是我们自己编写的代码。其中，`__keyevent@0__:expired` 就是一个默认的 channel，负责监听 key 的过期事件。也就是说，当一个 key 过期之后，Redis 会发布一个 key 过期的事件到 `__keyevent@<db>__:expired` 这个 channel 中。

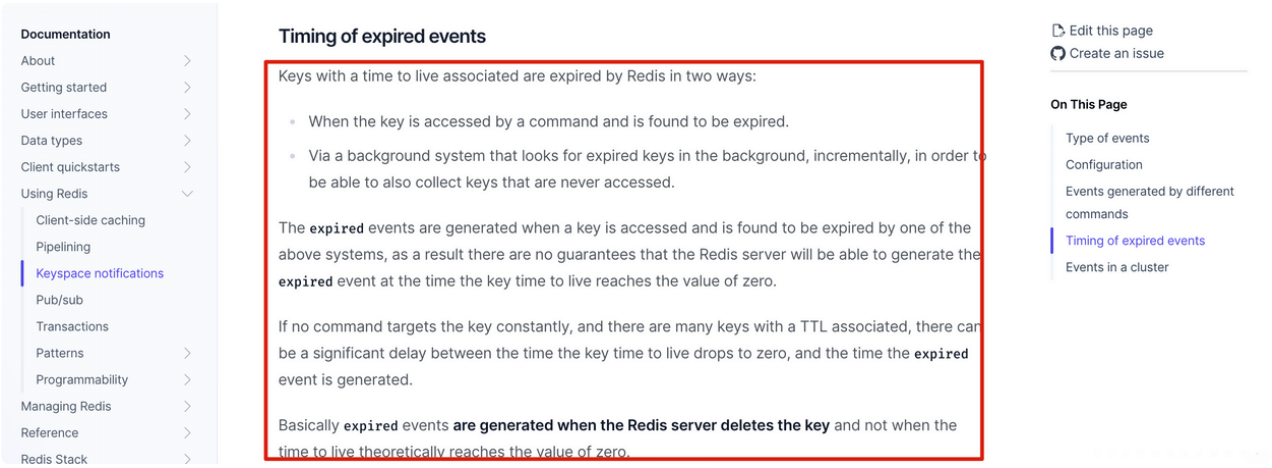
我们只需要监听这个 channel，就可以拿到过期的 key 的消息，进而实现了延时任务功能。

这个功能被 Redis 官方称为 **keyspace notifications** ，作用是实时监控实时监控 Redis 键和值的变化。

Redis 过期事件监听实现延时任务功能有什么缺陷?

1、时效性差

官方文档的一段介绍解释了时效性差的原因，地址：https://redis.io/docs/manual/keyspace-notifications/#timing-of-expired-events <https://redis.io/docs/manual/keyspace-notifications/#timing-of-expired-events> 。



这段话的核心是：过期事件消息是在 Redis 服务器删除 key 时发布的，而不是一个 key 过期之后就会就会直接发布。

我们知道常用的过期数据的删除策略就两个：

1. **惰性删除**：只会在取出 key 的时候才对数据进行过期检查。这样对 CPU 最友好，但是可能会造成太多过期 key 没有被删除。
2. **定期删除**：每隔一段时间抽取一批 key 执行删除过期 key 操作。并且，Redis 底层会通过限制删除操作执行的时长和频率来减少删除操作对 CPU 时间的影响。

定期删除对内存更加友好，惰性删除对 CPU 更加友好。两者各有千秋，所以 Redis 采用的是 **定期删除+惰性/懒汉式删除** 。

因此，就会存在我设置了 key 的过期时间，但到了指定时间 key 还未被删除，进而没有发布过期事件的情况。

2、丢消息

Redis 的 pub/sub 模式中的消息并不支持持久化，这与消息队列不同。在 Redis 的 pub/sub 模式中，发布者将消息发送给指定的频道，订阅者监听相应的频道以接收消息。当没有订阅者时，消息会被直接丢弃，在 Redis 中不会存储该消息。

3、多服务实例下存在消息重复消息的问题

Redis 的 pub/sub 模式目前只有广播模式，这意味着当生产者向特定频道发布一条消息时，所有订阅相关频道的消费者都能够收到该消息。

这个时候，我们需要注意多个服务实例重复处理消息的问题，这会增加代码开发量和维护难度。

为什么最后选择 Redisson 内置的延时队列来实现延时任务?

Redisson 是一个开源的 Java 语言 Redis 客户端，提供了很多开箱即用的功能，比如多种分布式锁的实现、延时队列。

我们可以借助 Redisson 内置的延时队列 RDelayedQueue 来实现延时任务功能。

Redisson的延迟队列 RDelayedQueue 是基于Redis的SortedSet来实现的。SortedSet是一个有序集合，其中的每个元素都可以设置一个分数，代表该元素的权重。Redisson利用这一特性，将需要延迟执行的任务插入到SortedSet中，并给它们设置相应的过期时间作为分数。

Redisson使用 `zrangebyscore` 命令扫描SortedSet中过期的元素，然后将这些过期元素从 SortedSet 中移除，并将它们加入到就绪消息列表中。就绪消息列表是一个阻塞队列，有消息进入就会被监听到。这样做可以避免对整个SortedSet进行轮询，提高了执行效率。

相比于 Redis 过期事件监听实现延时任务功能，这种方式具备下面这些优势：

1. **减少了丢消息的可能**：DelayedQueue 中的消息会被持久化，即使Redis宕机了，根据持久化机制，也只可能丢失一点消息，影响不大。当然了，你也可以使用扫描数据库的方法作为补偿机制。
2. **消息不存在重复消费问题**：每个客户端都是从同一个目标队列中获取任务的，不存在重复消费的问题。

跟 Redisson 内置的延时队列相比，消息队列可以通过保障消息消费的可靠性、控制消息生产者和消费者的数量等手段来实现更高的吞吐量和更强的可靠性，实际项目中首选使用消息队列的延时消息这种方案。

在下面这篇文章中有详细提到 MQ 延时任务解决方案，推荐看看：

订单超时自动取消如何实现?

如果在用户在一定的时间（比如 24 小时）内没有付款，那么订单会自动被取消，这是电商和...

《后端面试高频系统设计&场景题》

url=https%3A%2F%2Fwww.yuque.com%2Fsnailclimb%2Ftangw3%2Fdq6dco1estpldhv7&pic=nu