二

31 实战：定时任务案例

我在开发的时候曾经遇到了这样一个问题，产品要求给每个在线预约看病的患者，距离预约时间的前一天发送一条提醒推送，以防止患者错过看病的时间。这个时候就要求我们给每个人设置一个定时任务，用前面文章说的延迟队列也可以实现，但延迟队列的实现方式需要开启一个无限循环任务，那有没有其他的实现方式呢？

答案是肯定的，接下来我们就用 Keyspace Notifications（键空间通知）来实现定时任务，**定时任务指的是指定一个时间来执行某个任务，就叫做定时任务。**

开启键空间通知

默认情况下 Redis 服务器端是不开启键空间通知的，需要我们手动开启。

键空间开启分为两种方式：

- 命令设置方式
- 配置文件设置方式

接下来，我们分别来看。

命令设置方式

使用 redis-cli 连接到服务器端之后，输入 `config set notify-keyspace-events Ex` 命令，可以直接开启键空间通知功能，返回“OK”则表示开启成功，如下命令所示：

```
127.0.0.1:6379> config set notify-keyspace-events Ex
OK
```

优点：

- 设置方便，无序启动 Redis 服务。

缺点：

- 这种方式设置的配置信息是存储在内存中的，重启 Redis 服务之后，配置项会丢失。

配置文件设置方式

找到 Redis 的配置文件 redis.conf，设置配置项 `notify-keyspace-events Ex` ，然后重启 Redis 服务器。

优点：

- 无论 Redis 服务器重启多少次，配置都不会丢失。

缺点：

- 需要重启 Redis 服务。

配置说明

可以看出无论是那种方式，都是设置 notify-keyspace-events Ex，其中 Ex 表示开启键事件通知里面的 key 过期事件。

更多配置项说明如下：

- K：键空间通知，所有通知以 `__keyspace@<db>__` 为前缀
- E：键事件通知，所有通知以 `__keyevent@<db>__` 为前缀
- g：DEL、EXPIRE、RENAME 等类型无关的通用命令的通知
- $：字符串命令的通知
- l：列表命令的通知
- s：集合命令的通知
- h：哈希命令的通知
- z：有序集合命令的通知
- x：过期事件，每当有过期键被删除时发送
- e：驱逐（evict）事件，每当有键因为 maxmemory 政策而被删除时发送
- A：参数 g$lshzxe 的别名

以上配置项可以自由组合，例如我们订阅列表事件就是 El，但需要注意的是，**如果 notify-**

keyspace-event 的值设置为空，则表示不开启任何通知，有值则表示开启通知。

功能实现

我们要实现定时任务需要使用 Pub/Sub 订阅者和发布者的功能，使用订阅者订阅元素的过期事件，然后再执行固定的任务，这就是定时任务的实现思路。

以本文开头的问题为例，我们是这样实现此定时任务的，首先根据每个患者预约的时间往前推一天，然后再计算出当前时间和目标时间（预约前一天的时间）的毫秒值，把这个值作为元素的过期时间设置到 Redis 中，当这个键过期的时候，我们使用订阅者模式就可以订阅到此信息，然后再发提醒消息给此用户，这样就实现了给每个患者开启一个单独的分布式定时任务的功能。

我们先用命令的模式来模拟一下此功能的实现，首先，我们使用 redis-cli 开启一个客户端，监听 `__keyevent@0__:expired` 键过期事件，此监听值 `__keyevent@0__:expired` 为固定的写法，其中 0 表示第一个数据库，我们知道 Redis 中一共有 16 个数据，默认使用的是第 0 个，我们建议新开一个非 0 的数据库专门用来实现定时任务，这样就可以避免很多无效的事件监听。

命令监听如下：

```
127.0.0.1:6379> psubscribe __keyevent@0__:expired
1) "psubscribe"
2) "__keyevent@0__:expired"
3) (integer) 1
```

此时我们开启另一个客户端，添加两条测试数据试试，命令如下：

```
127.0.0.1:6379> set key value ex 3
OK
127.0.0.1:6379> set user xiaoming ex 3
OK
```

等过去 3 秒钟之后，我们去看监听结果如下：

```
127.0.0.1:6379> psubscribe __keyevent@0__:expired
1) "psubscribe"
2) "__keyevent@0__:expired"
3) (integer) 1
1) "pmessage"
2) "__keyevent@0__:expired"
```

```
3) "__keyevent@0__:expired"
4) "key" #接收到过期信息 key
1) "pmessage"
2) "__keyevent@0__:expired"
3) "__keyevent@0__:expired"
4) "user" #接收到过期信息 user
```

已经成功的介绍到两条过期信息了。

代码实战

本文我们使用 Jedis 来实现定时任务，代码如下：

```
import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPubSub;
import utils.JedisUtils;

/**
 * 定时任务
 */
public class TaskExample {
    public static final String _TOPIC = "__keyevent@0__:expired"; // 订阅频道名称
    public static void main(String[] args) {
        Jedis jedis = JedisUtils.getJedis();
        // 执行定时任务
        doTask(jedis);
    }

    /**
     * 订阅过期消息，执行定时任务
     * @param jedis Redis 客户端
     */
    public static void doTask(Jedis jedis) {
        // 订阅过期消息
        jedis.psubscribe(new JedisPubSub() {
            @Override
            public void onPMessage(String pattern, String channel, String message)
                // 接收到消息，执行定时任务
                System.out.println("收到消息：" + message);
            }
        }, _TOPIC);
    }
}
```

小结

本文我们通过开启 Keyspace Notifications 和 Pub/Sub 消息订阅的方式，可以拿到每个键

值过期的事件，我们利用这个机制实现了给每个人开启一个定时任务的功能，过期事件中我们可以获取到过期键的 key 值，在 key 值中我们可以存储每个用户的 id，例如“user_1001”的方式，其中数字部分表示用户的编号，通过此编号就可以完成给对应人发送消息通知的功能。

上一页 下一页