时间轮

版本	作者	描述	时间
1.0	benjamin	1.初稿	2017.07.27

工作原理

Redis 从2.8.0版本后,推出 Keyspace Notifications 特性后,Key过期能触发一个事件,然后通过扑捉该事件的方式对业务进行处理。

所需环境

```
redis 2.8.0
redis配置文件的参数notify-keyspace-events必须配置为"Ex"
```

代码

php服务端

```
* http://www.aweb.cc/guide/redis/notifications.html
* http://www.imooc.com/article/10431
* @desc: 时间轮
        工作原理: 利用redis数据过期发送消息机制,实现定时任务
* @author: benjamin
* @date: 2016年11月17日
* ===========tp3.1========
* @example: Vendor("TimeWheel#class");
           $timewheelObj = new TimeWheel();
           $timewheelObj->subscribe(function($instance, $channelName, $message){
                                           这里写业务逻辑
                ob_flush();//如果有输出的话,这里一定要写
           });
* ===============tp5.0、3.2=============
* import("Vendor.TimeWheel");
* #第一步,在需要定时处理的位置,设置key,下面的例子表示10分钟后处理
  $timeWheelObj = new \TimeWheel();
  $timeWheelObj->setRedis("travel:users:id:1:changeSex", "这是个测试", 600);
  #第二步,定时处理
  $timeWheelObj = new \TimeWheel();
  $timewheelObj->subscribe(function($instance, $channelName, $message){
    error_log(print_r($message,true),3,'timewheel.log');
                ob_flush();//如果有输出的话,这里一定要写
          });
class TimeWheelConfig
               = '116.62.148.118';
   const IP
   const PORT = 6379;
```

```
const CHANNEL = '__keyevent@0__:expired';
   const AUTH = "yuechain!@#$";
class TimeWheel
   protected $_redisObj;
   protected $_channel;
   * @desc
   * @access
   * @param string $ip redis的ip地址
           number $port redis的端口号
           string $channel redis消息通道
   * @return
   * @example
   * @date 2016年11月17日
   * @author benjamin
   function __construct($ip=TimeWheelConfig::IP,$port=TimeWheelConfig::PORT,$channel=TimeWheelConfig::CHANNEL,$auth=TimeWheelConfig::All
      ini_set('default_socket_timeout', -1);
       $this-> redisObj = new Redis();
      $this->_redisObj->connect($ip, $port);
      $this->_redisObj->auth($auth);
       $this-> channel = $channel;
   }
   * @desc 业务逻辑回调
   * @access
   * @param string $callback 匿名回调函数
   * @return
   * @example
   * @date 2016年11月17日
   * @author benjamin
   * @memo
   * 注意点一: callback回调函数有固定格式,格式如下所示
   * function($instance, $channelName, $message){
                //$instance 是redis实例,系统自动传递该参数,无需手动设置
                //$channelName 是消息通道名字
                //$message 是redies的key
                //所有业务逻辑围绕message展开
   *注意点二: message设计参考如下
   * 数据库名:表名:unique字段名:unique字段值:其他需要处理的业务逻辑
   */
   public function subscribe($callback)
       $this->_redisObj->subscribe(array($this->_channel), $callback);
   }
   /**
   * @desc
   * @param string $key key
   * @param string $value value
   * @param string $expire 超时时间,单位秒
```

```
* @return
* @example
* @date 2017年3月28日
* @author benjamin
*/
public function setRedis($key,$value,$expire)
{
$this->_redisObj->set("timewheel:".$key,$value,$expire);
}
```

php客户端

```
import("Vendor.TimeWheel.TimeWheel");
    $timeWheelObj = new \TimeWheel();
    $timeWheelObj->subscribe(function($instance, $channelName, $message){

// 这里写业务逻辑
    ob_flush();//如果有输出的话,这里一定要写
    });
```

shell脚本

```
#文件名base-deamon-timeWheel.sh
#!/bin/sh
while true;
do
    count=$(ps -ef | grep -c Wheel) #查找当前的进程中,计算server程序的数量
    if [ $count -lt 4 ]; then  #判断服务器进程的数量是否小于3(根据实际填上你的服务器进程数量)
    cd /www/shanpiao.tao3w.com
php index.php Api Shanpiao timeWheel &  #这里填入需要重启的服务器进程
fi
    sleep 2  #睡眠2s,周期性地检测服务器程序是不是崩溃了
done
```

运行shell脚本

```
[root@iZ23yrxtsktZ ~]# nohup bash ./base-deamon-timeWheel.sh &
```

参考

- http://www.imooc.com/article/10431
- https://my.oschina.net/itblog/blog/601284