# 总体来说消息队列的组成三部分:



## 主要用于:

## 大数据的短信发送,邮件发送,消息推送的,等耗时的任务异步的消息队列.

## 常用的消息队列

### Kafka(目前最火)

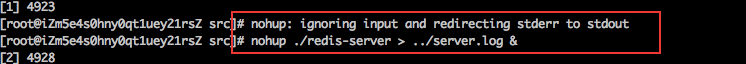
### Rabbitmq

### Redis

## 安装redis 服务:



**以后台的方式执行redis,并且记录下日志**



### 下载phpredis

1. wget <http://pecl.php.net/get/redis-5.0.2.tgz>
2. tar zxf redis-5.0.2.tgz
3. cd redis-5.0.2
4. /usr/local/php/bin/phpize
5. /configure --with-php-config=/usr/local/php/bin/php-config
6. Make && make install
7. vim /etc/php.ini
8. Extension=redis
9. 测试是否可以连接成功

public function redis(){

$redis = new \Redis();

$redis->connect("127.0.0.1",6379,5);

$redis->set('age',18);

return $this->writeJson(200,'ok',$redis->get("age"));

}

### 使用 redis的list 来实现redis

Rpush lpush 抛入(生产)

Rpop lpop 抛出(消费)

### 生产者

/\*\*

\* 生成者

\*

\* @return void

\*/

public function pub(){

$param = $this->request()->getRequestParam();

//抛入数据

Di::getInstance()->get('REDIS')->rPush('name\_list',$param['name']);

}

### 消费者

在全局服务下,开启三个进程,去执行任务

public static function mainServerCreate(EventRegister $register)

{

//注册redis进容器

Di::getInstance()->set('REDIS',Redis::getInstance());

$allNum = 3;

for ($i = 0 ;$i < $allNum;$i++){

//开启了是哪个子进程,自定义 进程名 , 去调用 ConsumerTest 里面的方法

ServerManager::getInstance()->getSwooleServer()->addProcess((new ConsumerTest("my\_dingyi\_consumer\_{$i}"))->getProcess());

}

}

要执行的任务,自动执行.

namespace App\Lib\Process;

use EasySwoole\Component\Process\AbstractProcess;

use Swoole\Process;

use EasySwoole\Component\Di;

class ConsumerTest extends AbstractProcess

{

private $isRun = false;

public function run($arg)

{

// TODO: Implement run() method.

/\*

\* 举例，消费redis中的队列数据

\* 定时500ms检测有没有任务，有的话就while死循环执行

\*/

$this->addTick(500,function (){

if(!$this->isRun){

$this->isRun = true;

//$redis = new \redis();//此处为伪代码，请自己建立连接或者维护redis连接

while (true){

try{

//容器取出

$task = Di::getInstance()->get('REDIS')->lPop('name\_list');

if($task){

// 发短信,发邮件,发消息的逻辑,写日志log

}else{

break;

}

}catch (\Throwable $throwable){

break;

}

}

$this->isRun = false;

}

var\_dump($this->getProcessName().' task run check');

});

}

public function onShutDown()

{

// TODO: Implement onShutDown() method.

}

public function onReceive(string $str, ...$args)

{

// TODO: Implement onReceive() method.

}

}