

東方通 | *TongTech*[®]
400-650-7088



TongLink/Q9.0

Java 客户端编程手册

版本: V9.0.2



目 录

1. 概述.....	1
1.1 目的.....	1
1.2 预期读者.....	1
2. 应用编写中用到的数据结构.....	1
2.1 概念.....	1
2.2 数据结构.....	1
2.2.1 Message 数据结构.....	3
2.2.2 Filemessage 数据结构.....	3
2.2.3 SendResult 数据结构.....	3
2.2.4 SendFileResult 数据结构.....	4
2.2.5 SendBatchResult 数据结构.....	4
2.2.6 DownloadReslut 数据结构.....	4
2.2.7 PullResult 数据结构.....	5
2.2.8 TLQLightTopicProducer#producerSend 数据结构.....	6
2.2.9 TLQTopicProducer#producerSendMessageAsync 数据结构.....	6
2.2.10 TLQLightTopicPullConsumer#pullMessage 数据结构.....	6
2.2.11 TLQTopicPullConsumer#pullMessageAsync 数据结构.....	7
2.2.12 TLQLightPairProducer#producerSend 数据结构.....	7
2.2.13 TLQLightPairPullConsumer#pullMessage 数据结构.....	8
2.2.14 TLQPairProducer#producerSendMessageAsync 数据结构.....	8
2.2.15 TLQPairPullConsumer#pullMessageAsync 数据结构.....	8
3. 类定义.....	9
3.1 胖客户端应用的类.....	9
3.1.1 TLQTopicProducer 生产者类.....	9
3.1.2 TLQTopicPullConsumer 消费者类.....	10
3.1.3 TLQTopicPushConsumer 消费者类.....	12
3.1.4 TLQPairProducer 生产者类.....	13
3.1.5 TLQPairPushConsumer 消费者类.....	15
3.1.6 TLQPairPullConsumer 消费者类.....	16
3.2 瘦客户端应用的类.....	17
3.2.1 TLQLightProducer 生产者类.....	17
3.2.2 TLQLightTopicPullConsumer 消费者类.....	18
3.2.3 TLQLightPairProducer 生产者类.....	19
3.2.4 TLQLightPairPullConsumer 消费者类.....	20
3.3 Message 消息.....	21
3.4 FileMessage 消息.....	22
4. 方法定义.....	23
4.1 方法声明.....	23
4.2 基本应用方法.....	23
4.2.1 Buff 消息相关方法.....	23
4.2.2 文件 BUFF 消息相关方法.....	26

4.2.3 客户端生产者发送/消费者拉取消息回调方法	27
4.3 胖客户端应用方法	27
4.3.1 发布订阅模式生产者相关方法	28
4.3.2 发布订阅模式消费者相关方法	30
4.3.3 点对点模式生产者相关方法	35
4.3.4 点对点模式消费者相关方法	38
4.4 瘦客户端相关方法	42
4.4.1 发布订阅模式生产者相关方法	42
4.4.2 发布订阅模式消费者相关方法	44
4.4.3 点对点模式生产者相关方法	47
4.4.4 点对点模式消费者相关方法	49
附录 1 错误编码信息	52

1. 概述

1.1 目的

本文档用于描述 TLQ9 java 客户端应用编写中用到的相关数据结构及方法接口定义等。通过阅读此手册能快速查找到应用编写中用到的方法及用法等。

1.2 预期读者

本接口设计说明书的预期读者是消息中间件 TLQ9 版本产品用户以及对消息中间件感兴趣的读者。

2. 应用编写中用到的数据结构

应用编写中用到的概念和数据结构有如下。

2.1 概念

同步模式客户端生产者：客户端发送同步消息机制，即当客户端发送消息成功后必须等到对应消息的确认消息才能继续发下一条消息；

异步模式客户端生产者：客户端发送异步消息机制，即当客户端发送消息不用等到消息发送完就可以继续发送下一条消息；

同步模式客户端消费者：客户端拉取同步消息机制，即当客户端发送拉取消息的请求之后需要等待返回的消息数据结果；

异步模式客户端消费者：客户端拉取异步消息机制，即当客户端注册异步回调函数后在消费者注销之前都可以拉取消息数据；

BUFF 消息：指内存中一段用户数据，包括消息头和消息内容，消息内容最大长度 4M；

批量消息：一种包含多个 BUFF 消息的数据结构；

管理节点：详细设计请参看系统使用手册。

2.2 数据结构

数据结构名称	含义
--------	----

Message	消息结构体
FileMessage	文件消息结构体
SendResult	发送消息响应结构体
SendBatchResult	发送批量消息响应结构体
SendFileResult	发送文件响应结构体
DownloadResult	下载文件响应结构体
PullResult	拉取消息响应结构体
TLQLightTopicProducer#producerSend	同步模式客户端生产者实例，发布订阅模式
TLQTopicProducer#producerSendMessageAsync	异步模式客户端生产者实例，发布订阅模式
TLQLightTopicPullConsumer#pullMessage	同步模式客户端消费者实例，发布订阅模式
TLQTopicPullConsumer#pullMessageAsync	异步模式客户端消费者实例，发布订阅模式
TLQLightPairProducer#producerSend	同步模式客户端生产者实例，点对点模式
TLQLightPairPullConsumer#pullMessage	同步模式客户端消费者实例，点对点模式
TLQPairProducer#producerSendMessageAsync	异步模式客户端生产者实例，点对点模式
TLQPairPullConsumer#pullMessageAsync	异步模式客户端消费者实例，点对点模式

2.2.1 Message 数据结构

属性	类型	含义	初始值
EXPIRY	String	消息的生命周期(-1 为不超时)	-1
PERSISTENCE	String	消息的持久化属性(0 非持久 1 持久)	0
PRIORITY	String	消息的优先级 0 至 9	0
topicOrQueue	String	主题信息或者队列信息	Null
properties	Map<String,String>	消息属性	Null
attr	Map<String, Object>	消息的自定义属性	Null
body	byte[]	消息的内容	Null
transactionId	String	传输消息 id	Null
msgId	String	消息的唯一 id	Null

备注：客户端在发送消息之前，将发送的消息构建成 Message 数据结构，进行发送。

2.2.2 Filemessage 数据结构

属性	类型	含义	初始值
splitFizeSize	long	大文件分片的大小	200M
EXPIRY	String	消息的生命周期(-1 为不超时)	-1
PERSISTENCE	String	消息的持久化属性(0 非持久 1 持久)	0
PRIORITY	String	消息的优先级 0 至 9	0
topicOrQueue	String	主题信息或者队列信息	Null
properties	Map<String,String>	消息属性	Null
attr	Map<String, Object>	消息的自定义属性	Null
body	byte[]	消息的内容	Null
transactionId	String	传输消息 id	Null
msgId	String	消息的唯一 id	Null

备注：客户端在发送消息之前，将发送的文件消息构建成 Filemessage 数据结构，进行发送。

2.2.3 SendResult 数据结构

属性	类型	含义	初始值
splitFizeSize	long	大文件分片的大小	200M
EXPIRY	String	消息的生命周期(-1 为不	-1

		超时)	
PERSISTENCE	String	消息的持久化属性(0 非持久 1 持久)	0
PRIORITY	String	消息的优先级 0 至 9	0
topicOrQueue	String	主题信息或者队列信息	Null
properties	Map<String,String>	消息属性	Null
attr	Map<String, Object>	消息的自定义属性	Null
body	byte[]	消息的内容	Null
transactionId	String	传输消息 id	Null
msgId	String	消息的唯一 id	Null

备注：客户端发送消息之后，将服务端收到的响应信息采用 **SendResult** 数据结构进行封装。

2.2.4 SendFileResult 数据结构

属性	类型	含义	初始值
sendStatus	SendStatus	发送消息状态【0 表示成功、1 表示失败】	0
msgId	String	消息 id	Null
fileId	long	文件 id	Null

备注：客户端发送文件之后，将服务端收到的响应信息采用 **SendFileResult** 数据结构进行封装。

2.2.5 SendBatchResult 数据结构

属性	类型	含义	初始值
sendStatus	SendStatus	发送消息状态【0 表示成功、1 表示失败】	0
batchId	String	批量消息 Id	Null
brokerId	String	工作节点 Id	Null

备注：客户端发送批量消息之后，将服务端收到的响应信息采用 **SendBatchResult** 数据结构进行封装。

2.2.6 DownloadResult 数据结构

属性	类型	含义	初始值
status	DownloadFileStatus	下载文件结果状态【DOWNLOAD_OK, DOWNLOAD_FAILED, DOWNLOAD	Null

		_FILE_NOT_EXITST】	
msgHeader	MessageHeader	文件头消息	Null
msgAttr	Map<String,Object>	文件消息自定义属性	Null
hash	String	文件哈希值	Null
fileName	String	文件名称	Null
realFileSize	long	文件的实际大小，文件压缩或者加密之后的文件大小	0
originalSize	long	文件进行压缩或加密之前的原始大小	0
fileId	long	文件 Id,大于 0	0
msgId	String	消息 id	Null
data	byte[]	文件数据信息	Null
minConsumeQueueOffset	long	批量消费记录最小偏移	0
maxConsumeQueueOffset	long	批量消费记录最大偏移	0
consumeHistoryOffset	long	服务端的消费记录偏移	0

备注：客户端从服务端下载文件时，将服务端收到的响应信息采用 DownloadResult 数据结构进行封装。

2.2.7 PullResult 数据结构

属性	类型	含义	初始值
pullStatus	PullStatus	拉取消息的状态【0 有消息、1 没有消息】	1
clientId	String	客户端实例 id	Null
consumerId	String	消费者 id	Null
groupName	String	消费者组名称	Null
topic	String	主题名称	Null
queueId	int	队列 id	0
minConsumeQueueOffset	long	批量消费记录最小偏移	0
maxConsumeQueueOffset	long	批量消费记录最大偏移	0
consumeHistoryOffset	long	服务端的消费记录偏移	0
statueCode	int	状态码	0
domain	String	域名	Null
msgFoundList	List<MessageExt>	消息集合	Null

备注：客户端从服务端拉取消息时，将服务端收到的响应信息采用 Pullresult 数据结构进行封装。

2.2.8 TLQLightTopicProducer#producerSend 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
topicName	String	主题名	Null
num	int	发送消息数量	10
persistence	int	消息持久性 0: 非持久性 1: 持久性	0
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布 2:全局发布	0
messageSize	int	发送每条消息大小	1024

备注：在发布订阅模式下瘦客户端生产者使用 TLQLightTopicProducer#producerSend 数据结构同步发送消息。

2.2.9 TLQTopicProducer#producerSendMessageAsync 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
topicName	String	主题名	Null
num	int	发送消息数量	10
persistence	int	消息持久性 0: 非持久性 1: 持久性	0
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布 2:全局发布	0
messageSize	int	发送每条消息大小	1024

备注：在发布订阅模式下胖客户端生产者使用 TLQTopicProducer#producerSendMessageAsync 数据结构异步发送消息。

2.2.10 TLQLightTopicPullConsumer#pullMessage 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
topicName	String	主题名	Null

pullType	PullType	拉取消息方式 >=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息	PullType.PullContinue
offset	long	拉取消息偏移量	0
maxNums	int	拉取消息最大数量	32
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布 2:全局发布	0
subscribeType	SubscribeType	订阅模式 0:持久订阅模式、1: 非持久持续订阅、2: 非持久订阅最新消息	SubscribeType.TLQ_SUB_DURABLE

备注：在发布订阅模式下瘦客户端消费者使用 TLQLightTopicPullConsumer#pullMessage 数据结构主动同步拉取消息。

2.2.11 TLQTopicPullConsumer#pullMessageAsync 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
queueName	String	主题名	Null
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布 2:全局发布	0

备注：在发布订阅模式下胖客户端消费者使用 TLQTopicPullConsumer#pullMessageAsync 数据结构主动异步拉取消息。

2.2.12 TLQLightPairProducer#producerSend 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
queueName	String	队列名	Null
num	int	发送消息数量	10
persistence	int	消息持久性 0: 非持久性 1: 持久性	0
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布	0

		布 2:全局发布	
messageSize	int	发送每条消息大小	1024
priority	int	消息优先级	1

备注：在队列模式下度客户端生产者使用 TLQLightPairProducer#producerSend 数据结构同步发送消息。

2.2.13 TLQLightPairPullConsumer#pullMessage 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
queueName	String	队列名	Null
useProxy	int	消息作用范围，0:本地发布 1:本地发布与全局发布 2:全局发布	0

备注：在队列模式下度客户端消费者使用 TLQLightPairPullConsumer#pullMessage 数据结构主动同步拉取消息。

2.2.14 TLQPairProducer#producerSendMessageAsync 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null
domain	String	域名	Null
queueName	String	队列名	Null
num	int	发送消息数量	10
persistence	int	消息持久性 0: 非持久性 1: 持久性	0
useProxy	int	消息作用范围，0:本地发布 1:本地发布与全局发布 2:全局发布	0
messageSize	int	发送每条消息大小	1024
priority	int	发送每条消息优先级	1

备注：在队列模式下胖客户端生产者使用 TLQPairProducer#producerSendMessageAsync 数据结构异步发送消息。

2.2.15 TLQPairPullConsumer#pullMessageAsync 数据结构

属性	类型	含义	初始值
nameSrv	String	管理节点 ip 地址和端口号	Null

domain	String	域名	Null
queueName	String	主题名	Null
useProxy	int	消息作用范围, 0:本地发布 1:本地发布与全局发布 2:全局发布	0
maxNums	int	拉取消息最大数量	32

备注：在队列模式下胖客户端消费者使用 TLQPairPullConsumer#pullMessageAsync 数据结构主动异步拉取消息。

3. 类定义

3.1 胖客户端应用的类

在 C/S 模式结构的应用中，在一个运行环境中多个实例共享一个连接发送消息。生产者和消费者采用异步方式操作消息。

3.1.1 TLQTopicProducer 生产者类

3.1.1.1 属性

TLQTopicProducer 中属性有：

SetNamesrvAddr (String)

生产者实例域名。

UserProxy (int)

生产者实例发送消息范围。

ProtocolType (ProtocolType)

生产者实例发送消息的传输协议，分为 ProtocolType.TCP 和 ProtocolType.UDP 两种方式。

breakPointTrans(BreakPointTrans)

生产者实例发送文件设置是否支持文件断点续传，包括 NO_SEQUEL 不续传，CONTINUATION 断点续传。

map (Map<String Object>)

消息的自定义属性，包括生命周期、持久性、优先级属性。

3.1.1.2 方法

start

语法： start()

功能： 初始化生产者后启动生产者。

举例： producer.start();

shutdown

语法： shutdown()

功能： 发送消息后使生产者客户端注销工作节点和管理节点，结束相关线程。
举例： `producer.shutdown();`

send

语法： `send(Message msg, SendCallback sendCallback, long timeout)`
参数： `msg`:消息实体，消息内容和自定义消息数据量最大 4M
`sendCallback`: 异步发送消息回调方法
`timeout`:超时时间，默认 10 秒
功能： 异步发送消息接口。
举例：

```
producer.send(msg, new SendCallback() {  
    public void onSuccess(SendResult sendResult) { downLatch.countDown();}  
    public void onException(Throwable e) {  
        downLatch.countDown();  
    }}, 10000);
```

sendFile

语法： `sendFile(FileMessage message, long timeout)`
参数： `message`:文件消息实体
`sendCallback`:回调函数
`timeout`:超时时间，默认 120000000 秒
功能： 异步发送文件接口。
举例：

```
producer.sendFile(message, new SendFileCallback() {  
    public void onSuccess(SendFileResult sendResult) {  
        downLatch.countDown();  
    }  
    public void onException(Throwable e) {  
        downLatch.countDown();  
    }}, 120000000);
```

sendBatch

语法： `sendBatch(Collection<Message> messages, long timeout)`
功能： 异步批量发送消息接口。
参数： `messages`:消息集合
`timeout`:超时时间，默认 10 秒
`sendCallback`:异步回调函数
举例：

```
for (int i = 0; i < num; i++) {  
    producer.sendBatch(list, new SendBatchCallback() {  
        public void onSuccess(SendBatchResult sendResult) {  
            downLatch.countDown();  
        }  
    })  
}
```

3.1.2 TLQTopicPullConsumer 消费者类

3.1.2.1 属性

TlqTopicPullConsumer 中属性有：

subscribe (String)

消费者订阅消费主题。

SetNameSrvAddr (String)

消费者所属管理节点地址。。

UserProxy (int)

消费消息的范围。

subscribeType (SubscribeType)

订阅消息的类型，0 持久订阅;1 非持久持续订阅消息。

3.1.2.2 方法

start

语法: start()

功能: 初始化消费者后启动消费者。

举例: consumer.start();

shutdown

语法: shutdown()

功能: 消费消息后使消费者客户端注销工作节点和管理节点，结束相关线程。

举例: consumer.shutdown();

pullMessage

语法: pullMessage(PullType pullType, long consumerOffset, int maxNums, PullCallback pullCallback, long timeout)

参数: pullType: 拉取消息方式, >=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息

consumerOffset: 消费消息偏移量

maxNums: 拉取消息的最大数量，默认 10

pullCallback:拉取消息回调函数

timeout:超时时间，默认 10 秒

功能: 发布订阅模式主动异步拉取订阅消息。

举例: consumer.pullMessage(pullType, offset, maxnums, new PullCallback() {
public void onSuccess(PullResult pullResult) {
System.out.println("胖客户端 Topic 模式异步拉取消息返回:" + pullResult);
downLatch.countDown();}
public void onException(Throwable e) {
System.out.println("异常: " + e);
downLatch.countDown();
}}, 10000);

pullFileMessage

语法: pullFileMessage(String path,PullType pullType, long consumerOffset, long timeout, DownloadCallback callback)

参数: path: 拉取文件的地址

timeout:超时时间，默认 10 秒

callback:拉取文件回调函数

功能: 异步拉取文件接口。

举例:

```
consumer.pullFileMessage(path, pullType, offset, 2000000, new DownloadCallback() {  
    public void onSuccess(DownloadResult result) {  
        downLatch.countDown();  
    }  
    public void onException(Throwable e) {  
        downLatch.countDown();  
    }  
});
```

pullMessage

语法: `pullMessage(PullType pullType, long consumerOffset,int maxNums, long timeout)`

参数: `pullType`: 拉取消息方式, 拉取消息方式, ≥ 0 : `pullOffset` 使用客户端传递 `offset` 拉取消息、-1: `pullContinue` 使用服务端历史消费 `offset` 拉取消息、-2: `pullLatest` 拉取最新消息

`consumerOffset`: 消费消息偏移量

`maxNums`: 拉取消息的最大数量

`timeout`:超时时间, 默认 60 秒

功能: 消费者基于偏移量消息回溯返回消息接口。

举例:

```
PullResult list1 = consumer.pullMessage(PullType.PullOffset, consumeQueOffset, max  
Nums, 60000);
```

3.1.3 TLQTopicPushConsumer 消费者类

3.1.3.1 属性

TlqTopicPushConsumer 中属性有:

domain(String)

消费者所属的域。

subscribe (String)

消费者订阅消费主题。

SetNameSrvAddr (String)

消费者所属管理节点地址。

UserProxy (int)

消费消息的范围。

subscribeType (SubscribeType)

订阅消息的类型, 0 持久订阅;1 非持久持续订阅消息。

pullBatchSize (int)

消费者被动每批接收消息的数量。

pullType (PullType)

拉取消息方式, `PullOffset`:大于等于 0 整数表示, 使用客户端传递大于等于零的 `offset` 获取消息;
`PullContinue`:-1 表示, 使用服务端的消费历史记录获取消息; `PullLatest`:-2 表示, 拉取最新消息。

NextOffset (int)

下一次拉取消息的偏移量

3.1.3.2 方法

start

语法: start()

功能: 初始化消费者后启动消费者。

举例: consumer.start();

shutdown

语法: shutdown()

功能: 消费消息后使消费者客户端注销工作节点和管理节点, 结束相关线程。

举例: consumer.shutdown();

registerMessageListener

语法: registerMessageListener(MessageListener messageListener)

参数: messageListener: 并发消费的回调函数。

功能: 消费者被动并发接收消息推送。

举例: consumer.registerMessageListener(new MessageListenerConcurrently() {
public ConsumeConcurrentlyStatus consumeMessage(PullResult pullResult, ConsumeContext context) {
System.out.printf("当前线程: %s 胖客户端 Topic 模式 消息大小: %s 胖客户端 Topic 模式 消息体: %s %n", Thread.currentThread().getName(), pullResult.getMsgFoundList().size(), pullResult);return ConsumeConcurrentlyStatus.CONSUME_SUCCESS;}});

registerMessageListener

语法: registerMessageListener(MessageListener messageListener)

参数: messageListener: 顺序消费的回调函数。

功能: 消费者被动顺序接收消息推送。

举例: consumer.registerMessageListener(new MessageListenerOrderly() {
public ConsumeConcurrentlyStatus consumeMessage(PullResult pullResult, ConsumeContext context) {
return ConsumeConcurrentlyStatus.CONSUME_SUCCESS;}});

3.1.4 TLQPairProducer 生产者类

3.1.4.1 属性

TlqPairProducer 中属性有:

SetNameSrvAddr (String)

生产者实例域名。

UserProxy (int)

生产者实例发送消息范围。

breakPointTrans(BreakPointTrans)

生产者实例发送文件设置是否支持文件断点续传，包括 NO_SEQUEL 不续传，CONTINUATION 断点续传。

map (Map<String Object>)

消息的自定义属性，包括生命周期、持久性、优先级属性。

3.1.4.2 方法

start

语法: start()

功能: 初始化生产者后启动生产者。

举例: producer.start();

shutdown

语法: shutdown()

功能: 发送消息后使生产者客户端注销工作节点和管理节点，结束相关线程。

举例: producer.shutdown();

send

语法: send(Message msg, SendCallback sendCallback, long timeout)

参数: msg:消息实体

sendCallback:回调函数

timeout:超时时间，默认 20 秒

功能: 异步发送文件接口。

举例:

```
producer.send(msg, new SendCallback() {  
    public void onSuccess(SendResult sendResult) {  
        downLatch.countDown();  
    }  
    public void onException(Throwable e) {  
        downLatch.countDown();  
    }  
}, 20000);
```

sendFile

语法: sendFile(FileMessage message, SendFileCallback callback,long timeout)

参数: message:文件消息实体，消息和自定义属性消息总和最大 4M

callback:回调函数

timeout:超时时间,默认 30000 秒

功能: 异步发送文件接口。

举例:

```
producer.sendFile(message, new SendFileCallback() {  
    public void onSuccess(SendFileResult sendResult) {  
        long end = System.currentTimeMillis();  
        downLatch.countDown();  
    }  
    public void onException(Throwable e) {  
        downLatch.countDown();  
    }  
}, 30000000);
```

sendBatch

语法: `sendBatch(Collection<Message> messages, SendBatchCallback sendCallback, long timeout)`

参数: `messages`:消息集合

`timeout`:超时时间, 默认 10 秒

`sendCallback`:异步回调函数

功能: 异步发送批量消息接口。

举例:

```
for (int i = 0; i < num; i++) {  
    producer.sendBatch(list, new SendBatchCallback() {  
        public void onSuccess(SendBatchResult sendResult) {  
            downLatch.countDown();  
        }  
        public void onException(Throwable e) {  
            downLatch.countDown();  
        }  
    }, 10000);  
}
```

3.1.5 TLQPairPushConsumer 消费者类

3.1.5.1 属性

TlqTopicPushConsumer 中属性有:

domain(String)

消费者所属的域。

subscribe (String)

消费者订阅消费队列。

SetNamesrvAddr (String)

消费者所属管理节点地址。

UserProxy (int)

消费消息的范围。

3.1.5.2 方法

start

语法: `start()`

功能: 初始化消费者后启动消费者。

举例: `consumer.start();`

shutdown

语法: `shutdown()`

功能: 消费消息后使消费者客户端注销工作节点和管理节点, 结束相关线程。

举例: `consumer.shutdown();`

registerMessageListener

语法: `registerMessageListener(MessageListener messageListener)`

参数: messageListener: 并发消费的回调函数。
功能: 消费者被动并发接收消息推送。
举例:

```
consumer.registerMessageListener(new MessageListenerConcurrently() {  
    public ConsumeConcurrentlyStatus consumeMessage(PullResult pullResult, ConsumeConcurrentlyContext context) {  
        return ConsumeConcurrentlyStatus.CONSUME_SUCCESS;}});
```

registerMessageListener

语法: `registerMessageListener(MessageListener messageListener)`
参数: messageListener: 顺序消费的回调函数。
功能: 消费者被动顺序接收消息推送。
举例:

```
consumer.registerMessageListener(new MessageListenerOrderly() {  
    public ConsumeConcurrentlyStatus consumeMessage(PullResult pullResult, ConsumeConcurrentlyContext context) {  
        return ConsumeConcurrentlyStatus.CONSUME_SUCCESS;}});
```

3.1.6 TLQPairPullConsumer 消费者类

3.1.6.1 属性

TlqTopicPullConsumer 中属性有:

domain (String)

消费者所属的域。

subscribe (String)

消费者订阅消费主题。

SetNameSrvAddr (String)

消费者所属管理节点地址。。

UserProxy (int)

消费消息的范围。

subscribeType (SubscribeType)

订阅消息的类型, 0 持久订阅;1 非持久持续订阅消息。

3.1.6.2 方法

start

语法: `start()`

功能: 初始化消费者后启动消费者。

举例: `consumer.start();`

shutdown

语法: `shutdown()`

功能: 消费消息后使消费者客户端注销工作节点和管理节点, 结束相关线程。

举例: `consumer.shutdown();`

pullMessageAsync

语法: `pullMessage(PullCallback pullCallback, long timeout)`

参数: `pullCallback`:拉取消息回调函数

`timeout`:超时时间, 默认 10 秒

功能: 主动异步拉取消息接口。

举例:

```
consumer.pullMessage(new PullCallback() {  
    public void onSuccess(PullResult pullResult) {  
        System.out.println("队列模式异步拉取消息返回:" + pullResult);}  
    public void onException(Throwable e) {  
        System.out.println("异常: " + e);  
    }  
}, 10000);
```

pullFileMessage

语法: `pullFileMessage(String path, long timeout, DownloadCallback callback)`

参数: `path`: 拉取文件的地址

`timeout`:超时时间, 默认 12000 秒

`callback`:拉取文件回调函数

功能: 异步拉取文件接口。

举例:

```
consumer.pullFileMessage(path, 12000000, new DownloadCallback() {  
    public void onSuccess(DownloadResult result) {  
        downLatch.countDown();  
        System.out.println("队列模式下载文件完成:" + result);}  
    public void onException(Throwable e) {  
        downLatch.countDown();  
        System.out.println("对列模式下载文件异常:" + e);  
    }  
});
```

3.2 瘦客户端应用的类

3.2.1 TLQLightProducer 生产者类

3.2.1.1 属性

参考 [TLQTopicProducer 生产者类](#) 属性。

3.2.1.2 方法

start

语法: `start()`

功能: 初始化生产者后启动生产者。

举例: `producer.start();`

shutdown

语法: shutdown()
功能: 发送消息后使生产者客户端注销工作节点和管理节点, 结束相关线程。
举例: producer.shutdown();

send

语法: send(Message msg, long timeout)
参数: msg:消息实体, 消息和自定义属性消息总和最大 4M
timeout:超时时间, 默认 30 秒
功能: 同步发送消息接口。
举例: Message msg = TimerUtils.buildMessage(messageSize,topicName,"");
msg.setAttr(attr);
msg.setPersistence(persistence);
SendResult sendResult = producer.send(msg, 30000);

sendFile

语法: sendFile(FileMessage message,long timeout)
参数: message:文件消息实体, 消息和自定义属性消息总和最大 4M
timeout:超时时间, 默认 2000 秒
功能: 同步发送文件接口。
举例: long start = System.currentTimeMillis();
SendFileResult sendResult = producer.sendFile(message, 2000000);
System.out.println("瘦客户端 Topic 模式发送文件结果:" + sendResult);
long end = System.currentTimeMillis();
System.out.println("发送文件时间:" + (end - start) / 1000 + "秒");

sendBatch

语法: sendBatch(Collection<Message> messages, long timeout)
参数: msg:消息集合
timeout:超时时间
功能: 同步批量发送消息接口。
举例: for (int i = 0; i < num; i++) {
SendBatchResult sendBatchResult = producer.sendBatch(list, 10000);
System.out.println("瘦客户端 Topic 模式发送批量消息返回: " + i + " " +
sendBatchResult);}

3.2.2 TLQLightTopicPullConsumer 消费者类

3.2.2.1 属性

参考 [TLQTopicPullConsumer 消费者类](#)属性。

3.2.2.2 方法

start

语法: start()
功能: 初始化消费者后启动消费者。
举例: consumer.start();

shutdown

语法: shutdown()
功能: 消费消息后使消费者客户端注销工作节点和管理节点, 结束相关线程。
举例: consumer.shutdown();

pullMessage

语法: pullMessage(PullType pullType, long consumerOffset, int maxNums, long timeout)
参数: pullType: 拉取消息方式, >=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息
consumerOffset: 拉取消息偏移量, 默认 0
maxNums: 拉取消息的最大数量, 默认 10
timeout: 超时时间, 默认 30 秒
功能: 主动同步拉取消息。
举例: PullResult pullResult = consumer.pullMessage(pullType, offset, maxnums, 30000);

pullFileMessage

语法: pullFileMessage(String path, PullType pullType, long consumerOffset, long timeout)
参数: path: 拉取文件的地址
timeout: 超时时间, 默认 6000 秒
功能: 同步下载文件接口。
举例: DownloadResult downloadResult = consumer.pullFileMessage(path, pullType, offset, 6000000);
System.out.println("瘦客户端 Topic 模式下载文件完成:" + downloadResult);

3.2.3 TLQLightPairProducer 生产者类

3.2.3.1 属性

参考 [TLQPairProducer 生产者类](#) 属性

3.2.3.2 方法

start

语法: start()
功能: 初始化生产者后启动生产者。
举例: producer.start();

shutdown

语法: shutdown()

功能： 发送消息后使生产者客户端注销工作节点和管理节点，结束相关线程。

举例： `producer.shutdown();`

send

语法： `send(Message msg, long timeout)`

参数： `msg`:消息实体，消息和自定义属性消息总和最大 4M
`timeout`:超时时间，默认 20 秒

功能： `msg`:消息实体,消息和自定义属性消息总和最大 4M
`timeout`:超时时间,默认 20 秒

举例： `SendResult sendResult = producer.send(msg, 20000);`
`System.out.println("瘦客户端队列模式发送消息结果:" + sendResult);`

sendBatch

语法： `sendBatch(Collection<Message> messages, long timeout)`

参数： `msg`:消息集合，
`timeout`:超时时间，默认 20 秒

功能： 同步发送批量消息接口。

举例： `for (int i = 0; i < num; i++) {`
`SendBatchResult sendBatchResult = producer.sendBatch(list, 20000);`
`System.out.println("队列模式发送批量消息返回: " + i + " " + sendBatchResult);`
`}`

sendFile

语法： `sendFile(FileMessage message,long timeout)`

参数： `message`:文件消息实体
`timeout`:超时时间功能：

功能： 同步发送文件接口。

举例： `long start = System.currentTimeMillis();`
`SendFileResult sendResult = producer.sendFile(message, 2000000);`
`System.out.println("队列模式发送文件结果:" + sendResult);`
`long end = System.currentTimeMillis();`
`System.out.println("发送文件时间:" + (end - start) / 1000 + "秒");`

3.2.4 TLQLightPairPullConsumer 消费者类

3.2.4.1 属性

参考 [TLQPairPullConsumer 消费者类](#)属性。

3.2.4.2 方法

start

语法： `start()`

功能： 初始化消费者后启动消费者。

举例： `consumer.start();`

shutdown

语法： `shutdown()`

功能： 消费消息后使消费者客户端注销工作节点和管理节点，结束相关线程。

举例： `consumer.shutdown();`

pullMessage

语法： `pullMessage(long timeout)`

参数： `imeout`:超时时间，默认 30 秒

功能： 主动同步拉取消息。

举例： `PullResult pullResult = consumer.pullMessage(30000);`
`System.out.println("队列模式拉取消息返回:" + pullResult);`

pullFileMessage

语法： `pullFileMessage(String path, long timeout)`

参数： `path`: 拉取文件的地址

`timeout`: 超时时间，默认 6000 秒

功能： 同步拉取文件接口。

举例： `DownloadResult downloadResult = consumer.pullFileMessage(path, 6000000);`
`System.out.println("队列模式下载文件完成:" + downloadResult);`

3.3 Message 消息

在进行发送消费消息时，需要定义消息结构体。

3.3.1.1 属性

EXPIRY (String)

消息生命周期

PERSISTENCE (String)

消息持久性

PRIORITY (String)

消息的优先级，可以分为 0 到 9，在点对点模式发送的消息优先级越大越提前发送。

topicOrQueue (String)

用于标识消息主题或者队列名称。

properties (Map)

消息的相关属性，例如生命周期，持久性等。

attr(Map)

消息自定义属性。

body (Byte)

消息的数据。

transactionId (String)

发送消息的传输 id。

msgId (String)

消息 id。

3.3.1.2 方法

setPersistence

语法: setPersistence(int persistence)

功能: 设置消息的持久性。

举例:

```
Map<String, Object> map = new HashMap<>();
map.put("Tong", "Tongtech");
.....
msg.setPersistence(persistence);
.....
```

setAttr

语法: setAttr(Map<String, Object> attr)

功能: 设置消息的自定义属性。

举例:

```
Map<String, Object> map = new HashMap<>();
map.put("Tong", "Tongtech");
.....
msg.setAttr(map);
.....
```

setExpiry

语法: setExpiry(int expire)

功能: 设置消息的生命周期。

举例:

```
Map<String, Object> map = new HashMap<>();
map.put("Tong", "Tongtech");
.....
msg.setExpire(expiry);
.....
```

setPriority

语法: setPriority(int priority)

功能: 点对点模式中设置消息优先级。

举例:

```
Map<String, Object> map = new HashMap<>();
map.put("Tong", "Tongtech");
.....
msg.setPriority(priority);
.....
```

3.4 FileMessage 消息

客户端发送之前, 需要定义文件消息结构体, FileMessage 类继承 Message 类。这里只描述文件消息特有的属性和方法, 其他方法参考 [Message 消息](#)。

3.4.1.1 属性

file(File)

文件实例

splitFizeSize

大文件分片大小

3.4.1.2 方法

setSplitFizeSize

语法: **setSplitFizeSize(long splitFizeSize)**

功能: 设置文件分片大小。

参数: splitFizeSize: 文件分片大小。

举例:

```
public void setSplitFizeSize(long splitFizeSize) {  
    this.splitFizeSize = splitFizeSize;  
}
```

4. 方法定义

本部分将详细介绍有关基本应用接口方法, 主要从各方法功能、原型、位置、说明等几方面进行具体介绍。没有做特别说明的接口函数既适用于 UNIX 环境又适用于 Windows 环境。

4.1 方法声明

在 Java 编程环境下, 编写 Java 的应用程序时, 必须引入 Java 客户端的 tlq9client.jar。方法中参数的详细设置参考[类定义](#)中相应类方法参数。

4.2 基本应用方法

4.2.1 Buff 消息相关方法

4.2.1.1 Message#Message 消息结构体的初始化方法

〔功能〕 BUFF 消息结构体的初始化。

〔原型〕 `public Message(String topicOrQueue, byte[] body);`

〔参数说明〕

输入:

topicOrQueue: 发送消息的主题或者队列;

body: 消息内容。

输出:

无。

【返回值】

Message。

【使用说明】

初始化一条 BUFF 消息，在生产者发送 BUFF 消息之前需要调用此函数。

4.2.1.2 Message#setTopicOrQueue 设置消息主题或队列

【功能】设置 BUFF 消息主题。

【原型】public void setTopicOrQueue(String topicOrQueue);

【参数说明】

输入：

topicOrQueue : 需要发送的消息主题或者队列名称。

输出：

无。

【返回值】

无。

【使用说明】

设置消息的主题或队列名称，在初始化一条 BUFF 消息后调用。

4.2.1.3 Message#setBody 设置消息内容

【功能】设置 BUFF 消息内容。

【原型】public void setBody(byte[] body);

【参数说明】

输入：

body: 需要发送的消息内容。

输出：

无。

【返回值】

无。

【使用说明】

设置消息内容，在初始化一条 BUFF 消息后调用。

4.2.1.4 Message#setPersistence 设置消息持久性

【功能】设置 BUFF 消息是否为持久性消息。

【原型】public void setPersistence (int persistence);

【参数说明】

输入：

persistence: 0:非持久 1:持久。

输出：

无。

【返回值】

无。

【使用说明】

设置消息持久性，在初始化一条 BUFF 消息后调用。

4.2.1.5 Message#setAttr 设置消息自定义属性

【功能】设置 BUFF 消息自定义属性。

【原型】`public void setAttr(Map<String, Object> attr);`

【参数说明】

输入：

attr: 消息自定义属性。

输出：

无。

【返回值】

无。

【使用说明】

设置消息自定义属性，在初始化一条 BUFF 消息后调用。

4.2.1.6 Message#setExpiry 设置消息生命周期

【功能】设置 BUFF 消息生命周期。

【原型】`public void setExpiry(int expire);`

【参数说明】

输入：

expire: 消息生命周期，时间单位毫秒，-1 表示生命周期无限长。

输出：

无。

【返回值】

无。

【使用说明】

设置消息生命周期，在初始化一条 BUFF 消息后调用。

4.2.1.7 Message#setPriority 设置消息优先级

【功能】设置 BUFF 消息优先级。

【原型】`public void setPriority(int priority);`

【参数说明】

输入：

priority: 消息优先级 0-9。

输出：

无。

【返回值】

无。

【使用说明】

点对点模式设置消息优先级，在初始化一条 BUFF 消息后调用

4.2.2 文件 BUFF 消息相关方法

4.2.2.1 FileMessage#setFile 设置文件对象

〔功能〕 设置文件消息来源的文件。

〔原型〕 `public void setFile(File file);`

〔参数说明〕

输入:

file: 文件对象。

输出:

无。

〔返回值〕

无。

〔使用说明〕

在发送文件消息时，指定消息的文件对象。

4.2.2.2 FileMessage#setTopicOrQueue 设置文件消息主题或队列

参考 [Message#setTopicOrQueue 设置消息主题或队列](#) 方法。

4.2.2.3 FileMessage#setPersistence 设置文件消息持久性

参考 [Message#setPersistence 设置消息持久性](#) 方法。

4.2.2.4 FileMessage#setAttr 设置文件消息自定义属性

参考 [Message#setAttr 设置消息自定义属性](#) 方法。

4.2.2.5 FileMessage#setExpiry 设置文件消息生命周期

参考 [Message#setExpiry 设置消息生命周期](#) 方法。

4.2.2.6 FileMessage#setPriority 设置文件消息优先级

参考 [Message#setPriority 设置消息优先级](#) 方法。

4.2.2.7 FileMessage#setSplitFizeSize 设置文件消息分片大小

〔功能〕 设置文件消息的分片大小。

〔原型〕 `public void setSplitFizeSize(long splitFizeSize);`

〔参数说明〕

输入:

splitFizeSize: 文件分片大小。

输出:

无。

【返回值】

无。

【使用说明】

在发送文件时，若是大文件需要对文件分片进行传输，设置每片文件大小。

4.2.3 客户端生产者发送/消费者拉取消息回调方法

4.2.3.1 onSuccess

【功能】客户端生产者异步发送消息成功时的回调接口。

【原型】`public void onSuccess(SendResult sendResult);`

【参数说明】

输入：

`sendResult` 客户端消息异步发送成功后工作节点执行该回调函数传入的结果数据。

输出：

`sendResult` 消息发送结果。

【返回值】

无。

【使用说明】

客户端生产者发送异步消息成功时需要传入该回调函数接口，详见 [TLQTopicProducer#send 生产者异步发送消息](#)。

4.2.3.2 onException

【功能】客户端生产者异步发送消息失败时的回调函数接口。

【原型】`public void onException(Throwable e);`

【参数说明】

输入：

`e` 客户端消息发送异步发送失败后工作节点执行该回调函数传入的异常。

输出：

`e` 消息发送异常。

【返回值】

无。

【使用说明】

客户端生产者发送异步消息失败时需要传入该回调函数接口。

4.3 胖客户端应用方法

4.3.1 发布订阅模式生产者相关方法

4.3.1.1 TLQTopicProducer#start 生产者启动

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无。

输出:

有错误抛出异常信息。

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

1. 调用先后顺序为:

- 1) 初始化 TLQTopicProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置本地或全局方式拉取 userProxy;
- 5) 启动客户端 start。

4.3.1.2 TLQTopicProducer#send 生产者异步发送消息

【功能】生产者向 broker 节点异步发送消息函数。

【原型】public void send(Message msg, SendCallback sendCallback, long timeout);

【参数说明】

输入:

msg: 消息实体;

sendCallback: 异步回调函数;

timeout: 超时时间。

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于向 broker 节点异步发送消息。

2. 调用先后顺序为:

- 1) 初始化 TLQTopicProducer 对象;
- 2) 设置管理节点的 ip 和端口号;
- 3) 设置域名 setDomain;
- 4) 设置本地或全局方式拉取 userProxy;

- 5) 调用生产者启动函数 start;
- 6) 创建消息实例 msg;
- 7) 生产者异步发送 BUFF 消息 send

4.3.1.3 TLQTopicProducer#producerSendBatch 生产者异步发送批量消息

【功能】客户端生产者发送一条批量消息。

【原型】public void sendBatch(Collection<Message> messages, SendBatchCallback sendCallback, long timeout);

【参数说明】

输入:

message:	批量消息集合体;
sendCallback:	异步发送消息的回调函数;
timeout:	超时时间。

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

客户端生产者发送批量消息。

3. 调用先后顺序为:

- 1) 初始化 TLQTopicProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置全局或本地拉取方式 userProxy。
- 5) 启动生产者 start;
- 6) 创建批量消息实例 msg;
- 7) 生产者同步发送批量 BUFF 消息 sendBatch。

4.3.1.4 TLQTopicProducer#sendFile 生产者异步发送文件

【功能】客户端生产者异步发送文件消息。

【原型】public void sendFile(FileMessage message, SendFileCallback callback, long timeout);

【参数说明】

输入:

message:	文件消息;
callback:	异步发送文件回调函数;
timeout:	超时时间。

输出:

有错误抛出异常信息。

【返回值】

无。

【使用说明】

客户端生产者异步发送文件。

4. 调用先后顺序为:

- 1) 初始化 TLQTopicProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置是否支持断点续传 setBreakPointTrans;
- 5) 启动生产者 start;
- 6) 创建文件实例;
- 7) 创建 BUFF 文件消息并且赋值自定义属性;
- 8) 生产者异步发送文件 BUFF 消息 sendFile。

4.3.1.5 TLQTopicProducer#shutdown 关闭生产者

【功能】客户端关闭生产者。

【原型】public void shutdown();

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

5. 调用先后顺序为:

- 1) 初始化 TLQTopicProducer 象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 启动生产者 start;
- 5) 进行各种计划任务;
- 6) 关闭客户端生产者 shutdown。

4.3.2 发布订阅模式消费者相关方法

4.3.2.1 TLQTopicPullConsumer#start 客户端发布订阅模式 pull 方式启动消费者

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无;

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

6. 调用先后顺序为：

- 1) 初始化 TLQTopicPullConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置主题信息 setTopicName；
- 5) 设置全局或本地拉取方式 userProxy。
- 6) 启动客户端 start。

4.3.2.2 TLQTopicPullConsumer#pullMessage 主动异步拉取消息消费

【功能】消费者向 broker 节点进行异步拉取消息。

【原型】public void pullMessage(PullType pullType, long consumerOffset, int maxNums, PullCallback pullCallback, long timeout);

【参数说明】

输入：

pullType:	拉取消息方式，>=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息
consumerOffset:	消费偏移量；
maxNums:	拉取最大消息量；
pullCallback:	异步拉取消息回调函数；
timeout:	超时时间。

输出：

有错误抛出异常信息；

【返回值】

无

【使用说明】

消费者进行主动向 broker 节点异步拉取消息。

7. 调用先后顺序为：

- 1) 初始化 TLQTopicPullConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 订阅主题信息 subscribe；
- 4) 设置域名 setDomain；
- 5) 设置消息订阅类型 setSubscribeType；
- 6) 设置本地或全局方式拉取 userProxy；
- 7) 启动消费者 start；
- 8) 从 broker 端拉取消息，并接收返回结果 pullMessage。

4.3.2.3 TLQTopicPullConsumer#pullFileMessage 异步拉取文件

【功能】消费者向 broker 节点进行异步拉取文件。

【原型】 public void pullFileMessage(String path, long timeout, DownloadCallback callback);

【参数说明】

输入:

path: 文件位置;
timeout: 超时时间;
callback: 异步拉取文件回调函数。

输出:

有错误抛出异常信息;

【返回值】

无

【使用说明】

消费者进行主动向 broker 节点异步拉取文件。

8. 调用先后顺序为:

- 1) 初始化 TLQTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNameSrV;
- 3) 订阅主题信息 setSubscribe;
- 4) 设置域名 setDomain;
- 5) 启动消费者 start;
- 6) 从 broker 端拉取文件, 并通过回调函数返回拉取结果 pullFileMessage。

4.3.2.4 TLQTopicPullConsumer#pullMessage 基于偏移量消息回溯

【功能】 客户端主动拉取基于偏移量消息回溯。

【原型】 public PullResult pullMessage(PullType pullType, long consumerOffset,int maxNums, long timeout)

【参数说明】

输入:

pullType: 拉取消息方式, >=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息。

consumerOffset: 消费偏移量;
maxNums: 最大拉取消息数量;
timeout: 超时时间。

输出:

pullMessage 基于偏移量同步拉取消息结果。

【返回值】

PullResult 同步拉取消息结果。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

9. 调用先后顺序为:

- 1) 初始化 TLQTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置订阅主题信息 setTopicName;
- 4) 设置域名 setDomain;

- 5) 设置全局或本地拉取方式 useProxy。
- 6) 启动客户端 start。
- 7) 从 broker 端拉取消息，并接收返回结果 rollBackMessageToOffset。

4.3.2.5 TLQTopicPullConsumer#shutdown 主动拉取消息的消费者关闭

【功能】客户端关闭生产者。

【原型】public void shutdown();

【参数说明】

输入：

无。

输出：

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

10. 调用先后顺序为：

- 1) 初始化 TLQTopicPullConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置订阅主题信息 subscribe；
- 5) 启动生产者 start；
- 6) 进行各种计划任务；
- 7) 关闭客户端生产者 shutdown。

4.3.2.6 TLQTopicPushConsumer#start 消费者 Push 被动推送消费启动方法

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入：

无；

输出：

有错误抛出异常信息；

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

11. 调用先后顺序为：

- 1) 初始化 TLQTopicPushConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置订阅主题信息 subscribe；

- 5) 设置偏移量 `setNextOffset`;
- 6) 设置本地或全局方式拉取 `userProxy`;
- 7) 设置订阅方式 `setSubscribeType`;
- 8) 设置每次接收信息的数量 `setPullBatchSize`;
- 9) 启动客户端 `start`。

4.3.2.7 TLQTopicPushConsumer#registerMessageListener 被动拉取消息消费者注册拉取消息的回调函数

【功能】自动拉取消息的消费者注册拉取消息的回调函数接口。

【原型】`public void registerMessageListener(MessageListener messageListener);`

【参数说明】

输入:

`messageListener`: 消费消息的监听类。

输出:

无。

【返回值】

无。

【使用说明】

客户端消费者自动拉取消息需要注册这个回调函数。

12. 调用先后顺序为:

- 1) 初始化 `TLQTopicPushConsumer` 对象;
- 2) 设置管理节点的 ip 和端口号 `setNameSrvAddr`;
- 3) 订阅主题信息 `subscribe`;
- 4) 设置域名 `setDomain`;
- 5) 设置拉取偏移量 `setNextOffset`;
- 6) 设置全局代理节点或本地代理节点的标识 `useProxy`。
- 7) 设置消息订阅类型 `setSubscribeType`;
- 8) 启动消费者 `start`;
- 9) 客户端自动消费者注册拉取消息的监听回调函数 `registerMessageListener`。

4.3.2.8 TLQTopicPushConsumer#shutdown 客户端发布订阅模式 push 方式消费者关闭

【功能】客户端关闭生产者。

【原型】`public void shutdown();`

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

13. 调用先后顺序为：

- 1) 初始化 TLQTopicPushConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置订阅主题信息 subscribe；
- 5) 设置偏移量 setNextOffset；
- 6) 设置本地或全局方式拉取 userProxy；
- 7) 设置订阅方式 setSubscribeType；
- 8) 设置每次接收信息的数量 setPullBatchSize；
- 9) 启动客户端 start。
- 10) 进行各种计划任务；
- 11) 关闭客户端生产者 shutdown。

4.3.3 点对点模式生产者相关方法

4.3.3.1 TLQPairProducer#start 启动生产者

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入：

无；

输出：

有错误抛出异常信息；

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

14. 调用先后顺序为：

- 1) 初始化 TLQPairProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 启动客户端 start。

4.3.3.2 TLQPairProducer#send 生产者异步发送消息

【功能】生产者向 broker 节点异步发送消息函数。

【原型】public void send(Message msg, SendCallback sendCallback, long timeout);

【参数说明】

输入：

msg: 消息实体;
sendCallback: 异步回调函数;
timeout: 超时时间。

输出:

有错误抛出异常信息。

【返回值】

无。

【使用说明】

用于向 broker 节点异步发送消息。

15. 调用先后顺序为:

- 1) 初始化 TLQPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置本地或全局方式拉取 userProxy;
- 5) 必须先调用生产者启动函数 start;
- 6) 设置异步回调函数 SendCallback;
- 7) 在回调函数中处理消息发送结果 SendResult。

4.3.3.3 TLQPairProducer#producerSendFileAsync 生产者异步发送文件

【功能】客户端生产者异步发送文件。

【原型】public void sendFile(FileMessage message, SendFileCallback callback,long timeout);

【参数说明】

输入:

message: 文件消息;
callback: 发送文件的回调函数;
timeout: 超时时间。

输出:

sendFile: 文件发送消息。

【返回值】

无。

【使用说明】

客户端生产者发送文件消息。

16. 调用先后顺序为:

- 1) 初始化 TLQPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 创建文件实例 file 对象;
- 5) 创建 BUFF 文件消息;
- 6) 设置文件消息属性;
- 7) 启动生产者 start;
- 8) 创建批量消息实例 msg;
- 9) 生产者发送文件 BUFF 消息, 利用回调函数显示发送结果。

4.3.3.4 TLQPairProducer#sendBatch 客户端异步发送批量消息

〔功能〕 客户端生产者异步批量发送消息。

〔原型〕 `public void sendBatch(Collection<Message> messages, SendBatchCallback sendCallback, long timeout);`

〔参数说明〕

输入：

message: 批量消息集合;
callback: 发送文件的回调函数;
timeout: 超时时间。

输出：

sendFile: 批量发送消息。

〔返回值〕

无。

〔使用说明〕

客户端生产者发送文件消息。

17. 调用先后顺序为：

- 1) 初始化 TLQPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 启动生产者 start;
- 5) 创建批量消息实例 msg;
- 6) 生产者发送文件 BUFF 消息，利用回调函数显示发送结果。

4.3.3.5 TLQPairProducer#shutdown 关闭生产者

〔功能〕 客户端关闭生产者。

〔原型〕 `public void shutdown();`

〔参数说明〕

输入：

无。

输出：

无。

〔返回值〕

无。

〔使用说明〕

客户端关闭生产者。

18. 调用先后顺序为：

- 1) 初始化 TLQPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置全局或者本地发送方式 userProxy;
- 5) 启动生产者 start;

- 6) 进行各种计划任务;
- 7) 关闭客户端生产者 shutdown。

4.3.4 点对点模式消费者相关方法

4.3.4.1 TLQPairPullConsumer#start 启动消费者主动异步拉取消息方法

【功能】启动消费者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无;

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

19. 调用先后顺序为:

- 1) 初始化 TLQPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置本地或全局方式拉取消息 userProxy;
- 5) 启动客户端 start。

4.3.4.2 TLQPairPullConsumer#pullMessage 主动异步拉取消息方法

【功能】消费者向 broker 节点进行异步拉取消息。

【原型】public void pullMessage(PullCallback pullCallback, long timeout)

【参数说明】

输入:

pullCallback: 异步拉取消息回调函数;

timeout: 超时时间。

输出:

无;

【返回值】

无

【使用说明】

消费者进行向 broker 节点异步拉取消息。

20. 调用先后顺序为:

- 1) 初始化 TLQPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 订阅队列信息 subscribe;

- 4) 设置域名 setDomain;
- 5) 设置本地或全局方式拉取消息 userProxy;
- 6) 启动消费者 start;
- 7) 消费者拉取消息, 利用回调函数显示拉取结果信息。

4.3.4.3 TLQPairPullConsumer#pullFileMessage 主动异步方式拉取文件方法

【功能】消费者向 broker 节点进行异步拉取文件。

【原型】public void pullFileMessage(String path, long timeout, DownloadCallback callback);

【参数说明】

输入:

path: 文件的位置;
callback: 异步拉取文件的回调函数。

输出:

有错误抛出异常信息;

【返回值】

无

【使用说明】

消费者向 broker 节点异步拉取文件。

21. 调用先后顺序为:

- 1) 初始化 TLQPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNameSrV;
- 3) 设置订阅队列信息 setSubscribe;
- 4) 设置域名 setDomain;
- 5) 启动消费者 start;
- 6) 消费者获取 topic 路由信息 fetchSubscribeMessageQueues;
- 7) 从 broker 端拉取消息, 并回调函数接收返回结果。

4.3.4.4 TLQPairPullConsumer#shutdown 关闭主动拉取消息消费者方法

【功能】客户端关闭生产者。

【原型】public void shutdown();

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

22. 调用先后顺序为:

- 1) 初始化 TLQPairPullConsumer 象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;

- 3) 设置域名 setDomain;
- 4) 设置全局或本地拉取方式 userProxy。
- 5) 启动生产者 start;
- 6) 进行各种计划任务;
- 7) 关闭客户端生产者 shutdown。

4.3.4.5 TLQPairPushConsumer#start 启动 Push 被动消费消息消费者方法

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无;

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

23. 调用先后顺序为:

- 1) 初始化 TLQPairPushConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 设置本地或全局方式拉取消息 userProxy;
- 5) 启动客户端 start。

4.3.4.6 TLQPairPushConsumer#registerMessageListner Push 被动消费信息方法

【功能】客户端 push 方式并发获取信息。

【原型】public void registerMessageListener(MessageListener messageListener);

【参数说明】

输入:

messageListner: 并发拉取消息回调函数。

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动客户端 push 方式并发接收消息进行消费。

24. 调用先后顺序为:

- 1) 初始化 TLQPairPushConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置订阅队列信息;
- 4) 设置域名 setDomain;

- 5) 设置全局或本地拉取方式 userProxy。
- 6) 启动客户端 start
- 7) 消费者注册并发监听函数，利用回调函数返回消费信息。

4.3.4.7 TLQPairPushConsumer#shutdown 关闭 Push 被动消费信息消费者方法

【功能】客户端关闭生产者。

【原型】public void shutdown();

【参数说明】

输入：

无。

输出：

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

25. 调用先后顺序为：

- 1) 初始化 TLQPairPushConsumer 象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置全局或本地拉取方式 userProxy。
- 5) 启动生产者 start；
- 6) 进行各种计划任务；
- 7) 关闭客户端生产者 shutdown。

4.3.4.8 TLQLightTopicProducer#send 生产者同步发送消息方法

【功能】生产者向 broker 节点同步发送消息函数。

【原型】public SendResult send(Message msg, long timeout);

【参数说明】

输入：

msg: 消息实体；

timeout: 超时时间。

输出：

SendResult: 同步发送结果。

【返回值】

SendResult: 发送结果。

【使用说明】

用于向 broker 节点同步发送消息。

26. 调用先后顺序为：

- 1) 初始化 TLQLightTopicProducer 对象；
- 2) 设置管理节点的 ip 和端口号；
- 3) 设置域名 setDomain；

- 4) 设置主题消息;
- 5) 设置全局代理或本地节点代理标识 useProxy。
- 6) 调用生产者启动函数 start;
- 7) 创建消息实例 msg;
- 8) 生产者同步发送 BUFF 消息 send。

4.4 度客户端相关方法

4.4.1 发布订阅模式生产者相关方法

4.4.1.1 TLQLightTopicProducer#start 启动生产者方法

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无;

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

27. 调用先后顺序为:

- 1) 初始化 TLQLightProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 启动客户端 start。

4.4.1.2 TLQLightTopicProducer#sendFile 生产者同步发送文件方法

【功能】客户端生产者异步发送文件消息。

【原型】public void sendFile(FileMessage message, long timeout);

【参数说明】

输入:

message: 文件消息;

timeout: 超时时间。

输出:

有错误抛出异常信息。

【返回值】

无。

【使用说明】

客户端生产者同步发送文件。

28. 调用先后顺序为：

- 1) 初始化 TLQLightTopicProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置是否支持断点续传 setBreakPointTrans；
- 5) 启动生产者 start；
- 6) 创建文件实例；
- 7) 创建 BUFF 文件消息并且赋值自定义属性；
- 8) 生产者同步发送文件 BUFF 消息 sendFile。

4.4.1.3 TLQLightTopicProducer#sendBatch 生产者同步发送批量消息方法

【功能】客户端生产者发送一条批量消息。

【原型】public SendBatchResult sendBatch(Collection<Message> messages, long timeout)；

【参数说明】

输入：

message: 需要发送的批量消息集合；
timeout: 超时时间。

输出：

SendBatchResult: 同步发送消息结果。

【返回值】

SendBatchResult: 消息发送结果。

【使用说明】

客户端生产者发送批量消息。

29. 调用先后顺序为：

- 1) 初始化 TLQLightTopicProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置全局代理或本地节点代理标识 useProxy。
- 5) 启动生产者 start；
- 6) 创建批量消息实例 msg；
- 7) 生产者同步发送批量 BUFF 消息 sendBatch。

4.4.1.4 TLQLightTopicProducer#shutdown 关闭生产者方法

【功能】客户端关闭生产者。

【原型】public void shutdown()；

【参数说明】

输入：

无。

输出：

无。

【返回值】

无。

【使用说明】

客户端关闭生产者。

30. 调用先后顺序为：

- 1) 初始化 TLQLightTopicProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 启动生产者 start；
- 5) 进行各种计划任务；
- 6) 关闭客户端生产者 shutdown。

4.4.2 发布订阅模式消费者相关方法

4.4.2.1 TLQLightTopicPullConsumer#start 启动消费者方法

【功能】启动消费者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入：

无；

输出：

有错误抛出异常信息；

【返回值】

无。

【使用说明】

用于启动消费者和初始化客户端上下文信息。

31. 调用先后顺序为：

- 1) 初始化 TLQTopicPullConsumer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 订阅主题信息 subscribe；
- 4) 设置域名 setDomain；
- 5) 设置消息订阅类型 setSubscribeType；
- 6) 启动消费者 start。

4.4.2.2 TLQLightTopicPullConsumer#pullMessage 主动同步拉取消息方法

【功能】消费者向 broker 节点进行主动同步拉取消息。

【原型】public PullResult pullMessage(PullType pullType, long consumerOffset, int maxNums, long timeout);

【参数说明】

输入：

pullType: 拉取方式, >=0: pullOffset 使用客户端传递 offset 拉取消息、-1: pullContinue 使用服务端历史消费 offset 拉取消息、-2: pullLatest 拉取最新消息;
consumerOffset: 消费消息偏移量;
maxNums: 拉取消息条数;
timeout: 超时时间。

输出:

pullResult 同步拉取消息结果;

【返回值】

PullResult 拉取消息结果;

【使用说明】

消费者向 broker 节点持续拉取消息。

32. 调用先后顺序为:

- 1) 初始化 TLQLightTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 订阅主题信息 subscribe;
- 4) 设置域名 setDomain;
- 5) 设置消息订阅类型 setSubscribeType;
- 6) 启动消费者 start;
- 7) 从 broker 端拉取消息, 并接收返回结果 pullMessage。

4.4.2.3 TLQTopicPullConsumer#pullFileMessage 主动同步拉取文件方法

【功能】消费者向 broker 节点进行主动同步拉取文件。

【原型】public DownloadResult pullFileMessage(String path, long timeout);

【参数说明】

输入:

path: 拉取文件的地址;
timeout: 超时时间。

输出:

DownloadResult: 同步拉取文件结果;

【返回值】

DownloadResult: 拉取文件结果;

【使用说明】

消费者进向 broker 节点持续拉取文件。

33. 调用先后顺序为:

- 1) 初始化 TLQTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 启动消费者 start;
- 5) 从 broker 端拉取消息, 并接收返回结果 pullFileMessage。

4.4.2.4 TLQLightTopicPullConsumer#shutdown 关闭主动拉取消息消费者方法

【功能】客户端关闭消费者。

【原型】 public void shutdown();

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭主动消费者。

34. 调用先后顺序为:

- 1) 初始化 TLQLightTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 调用消费者启动函数 start;
- 5) 进行各种计划任务;
- 6) 关闭客户端消费者 shutdown。

4.4.2.5 TLQTopicPullConsumer#shutdown 关闭主动拉取消息消费者方法

【功能】 关闭自动拉取消息的消费者。

【原型】 public void shutdown();

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭自动消费者。

35. 调用先后顺序为:

- 1) 初始化 TLQTopicPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 订阅主题信息 subscribe;
- 4) 设置域名 setDomain;
- 5) 设置消息订阅类型 setSubscribeType;
- 6) 启动消费者 start;
- 7) 客户端消费者注册拉取消息的监听回调函数 registerMessageListener;
- 8) 关闭客户端自动消费者 shutdown。

4.4.3 点对点模式生产者相关方法

4.4.3.1 TLQLightPairProducer#start 生产者启动方法

【功能】启动生产者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入：

无；

输出：

有错误抛出异常信息；

【返回值】

无。

【使用说明】

用于启动生产者和初始化客户端上下文信息。

36. 调用先后顺序为：

- 1) 初始化 TLQLightPairProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置全局或本地发布模式 useProxy；
- 5) 启动客户端 start。

4.4.3.2 TLQLightPairProducer#send 同步发送消息方法

【功能】生产者向 broker 节点同步发送消息函数。

【原型】public SendResult send(Message msg,long timeout);

【参数说明】

输入：

msg 消息实体；

timeout 超时时间。

输出：

sendResult 同步发送结果。

【返回值】

sendResult 发送结果。

【使用说明】

用于向 broker 节点同步发送消息。

37. 调用先后顺序为：

- 1) 初始化 TLQLightPairProducer 对象；
- 2) 设置管理节点的 ip 和端口号；
- 3) 设置域名 setDomain；
- 4) 设置全局或本地发布模式 useProxy；
- 5) 调用生产者启动函数 start；
- 6) 设置消息自定义属性 Map<String,Object>;

- 7) 创建消息实例 msg，添加自定义属性 attr；
- 8) 生产者同步发送 BUFF 消息 send。

4.4.3.3 TLQLightPairProducer#sendBatch 同步发送批量消息方法

【功能】客户端生产者发送一条批量消息。

【原型】public SendBatchResult sendBatch(Collection<Message> messages, long timeout);

【参数说明】

输入：

message: 需要发送的批量消息数据；
timeout: 超时时间。

输出：

sendBatchResult 同步发送批量消息结果。

【返回值】

sendBatchResult 批量消息发送结果。

【使用说明】

客户端生产者发送批量消息。

38. 调用先后顺序为：

- 1) 初始化 TLQLightProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；
- 4) 设置全局或本地发布模式 useProxy；
- 5) 启动生产者 start；
- 6) 创建批量消息实例 msg；
- 7) 生产者同步发送批量 BUFF 消息 sendBatch。

4.4.3.4 TLQLightPairProducer#sendFile 同步发送文件消息方法

【功能】客户端生产者向 broker 同步发送文件。

【原型】public SendFileResult sendFile(FileMessage message, long timeout);

【参数说明】

输入：

message: 文件消息；
timeout: 超时时间；

输出：

sendFileResult: 同步发送文件消息结果

【返回值】

sendFileResult: 发送文件消息结果；

【使用说明】

客户端生产者同步发送文件消息。

39. 调用先后顺序为：

- 1) 初始化 TLQLightPairProducer 对象；
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr；
- 3) 设置域名 setDomain；

- 4) 设置生产者支持断点续传功能 setBreakPointTrans;
- 5) 启动生产者 start;
- 6) 创建文件实例 file;
- 7) 生成 BUFF 文件消息;
- 8) 生产者同步发送批量 BUFF 文件消息。

4.4.3.5 TLQLightPairProducer#shutdown 关闭生产者方法

【功能】关闭自动拉取消息的消费者。

【原型】public void shutdown();

【参数说明】

输入:

无。

输出:

无。

【返回值】

无。

【使用说明】

客户端关闭自动消费者。

40. 调用先后顺序为:

- 1) 初始化 TLQLightPairProducer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 设置域名 setDomain;
- 4) 启动消费者 start;
- 5) 进行各项计划任务;
- 6) 关闭客户端自动消费者 shutdown。

4.4.4 点对点模式消费者相关方法

4.4.4.1 TLQLightPairPullConsumer#start 启动主动同步方式消费消息消费者方法

【功能】启动消费者初始化客户端连接上下文信息。

【原型】void start();

【参数说明】

输入:

无;

输出:

有错误抛出异常信息;

【返回值】

无。

【使用说明】

用于启动消费者和初始化客户端上下文信息。

41. 调用先后顺序为:

- 1) 初始化 TLQLightPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNameSrvAddr;
- 3) 订阅队列信息 subscribe;
- 4) 设置域名 setDomain;
- 5) 设置全局或者本地发布模式 useProxy;
- 6) 启动消费者 start。

4.4.4.2 TLQLightPairPullConsumer#pullMessage 主动同步拉取消息方法

【功能】消费者向 broker 节点进行拉取消息。

【原型】public PullResult pullMessage(long timeout);

【参数说明】

输入:

timeout: 超时时间。

输出:

PullResult:同步拉取消息的结果;

【返回值】

PullResult:拉取文件的消息结果

【使用说明】

消费者进行向 broker 节点同步拉取消息。

42. 调用先后顺序为:

- 1) 初始化 TLQLightPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNameSrV;
- 3) 订阅队列信息 setSubscribe;
- 4) 设置域名 setDomain;
- 5) 设置全局或者本地发布模式 useProxy;
- 6) 启动消费者 start;
- 7) 从 broker 端拉取消息, 并接收返回结果 Pullresult。

4.4.4.3 TLQLightPairPullConsumer#pullFileMessage 主动同步拉取文件方法

【功能】消费者向 broker 节点进行拉取文件。

【原型】public DownloadResult pullFileMessage(String path, long timeout);

【参数说明】

输入:

path: 文件的位置;

timeout: 超时时间。

输出:

DownloadResult:同步拉取文件的消息;

【返回值】

Downloadresult:拉取文件的消息结果

【使用说明】

消费者进行向 broker 节点拉取文件消息。

43. 调用先后顺序为:

- 1) 初始化 TLQLightPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNameSrV;
- 3) 订阅队列信息 setSubscribe;
- 4) 设置域名 setDomain;
- 5) 设置全局或者本地发布模式 useProxy;
- 6) 启动消费者 start;
- 7) 从 broker 端拉取消息, 并接收返回结果 Downloadresult。

4.4.4.4 TLQLightPairPullConsumer#shutdown 关闭主动同步方式消费者方法

〔功能〕 关闭自动拉取消息的消费者。

〔原型〕 public void shutdown();

〔参数说明〕

输入:
无。

输出:
无。

〔返回值〕

无。

〔使用说明〕

客户端关闭消费者。

44. 调用先后顺序为:

- 1) 初始化 TLQLightPairPullConsumer 对象;
- 2) 设置管理节点的 ip 和端口号 setNamesrvAddr;
- 3) 订阅队列信息 subscribe;
- 4) 设置域名 setDomain;
- 5) 设置全局或者本地发布模式 useProxy
- 6) 启动消费者 start;
- 7) 客户端消费者拉取各项任务;
- 8) 关闭客户端消费者 shutdown。

附录 1 错误编码信息

列出目前会产生的错误编码信息：

错误信息	错误数值	错误原因
CONNECT_BROKER_EXCEPTION	-10001	连接 broker 异常
ACCESS_BROKER_TIMEOUT	-10002	与 broker 通信超时
BROKER_NOT_EXIST_EXCEPTION	-10003	broker 不存在
NO_NAME_SERVER_EXCEPTION	-10004	nameserver 不存在
NOT_FOUND_TOPIC_EXCEPTION	-10005	没有发现 topic
REQUEST_TIMEOUT_EXCEPTION	-10006	请求超时
CREATE_REPLY_MESSAGE_EXCEPTION	-10007	创建消息异常
MESSAGE_ILLEGAL	-10008	发送的消息不合理

