|  |  |  |  |
| --- | --- | --- | --- |
| **Broker配置参数** | | | |
| **#** | **参数名** | **默认值** | **说明** |
| 2 | listenPort | 10911 | broker的服务端口号，作为对producer和consumer使用服务的端口号 |
| 3 | namesrvAddr | null | namesrv的ip地址。格式: ip:port;ip:port |
| 4 | brokerIP1 | 本机IP | broker所在的机器ip，默认不用设置，如果机器有多个网卡，需要手动设置 |
| 5 | brokerName | 本机主机名 | 作用为一组master与slave通过brokerName是否相同来标示，通过brokerId来区分master还是slave |
| 6 | brokerClusterName | DefaultCluster | 整个broker集群的名字，创建topic时需要指定。 |
| 7 | brokerId | 0 | 0:master 非0:slave |
| 8 | storePathCommitLog | $HOME/store/commitlog/ | commitLog存储路径 |
| 9 | storePathConsumerQueue | $HOME/store/consumequeue/ | 消费队列存储路径 |
| 10 | mapedFileSizeCommitLog | 1024 \* 1024 \* 1024(1G) | commitLog每个文件的大小，默认1G |
| 11 | deleteWhen | 4 | 删除文件时间点，默认凌晨 4点 |
| 12 | fileReservedTime | 72 | 文件保留时间，默认72小时. |
| 13 | brokerRole | ASYNC\_MASTER | Broker 的角色  ASYNC\_MASTER 异步复制Master  SYNC\_MASTER 同步双写Master  SLAVE |
| 14 | flushDiskType | ASYNC\_FLUSH | 刷盘方式  ASYNC\_FLUSH 异步刷盘  SYNC\_FLUSH 同步刷盘 |
| 15 | defaultTopicQueueNums | 4 | 在发送消息时，自动创建服务器不存在的topic，默认创建的队列数。 |
| 16 | autoCreateTopicEnable | true | 是否自动创建topic。 |
| 17 | autoCreateSubscriptionGroup | true | 是否允许Broker自动创建订阅组，建议线下开启，线上关闭 |
| 18 | rejectTransactionMessage | false | 是否拒绝事务消息接入 |
| 19 | etchNamesrvAddrByAddressServer | false | 是否从web服务器获取Name Server地址，针对大规模的Broker集群建议使用这种方式 |
| 20 | storePathIndex | $HOME/store/index | 消息索引存储路径 |
| 21 | storeCheckpoint | $HOME/store/checkpoint | checkpoint文件存储路径 |
| 22 | abortFile | $HOME/store/abort | abort文件存储路径 |
| 23 | maxTransferBytesOnMessageInMemory | 262144 | 单次Pull消息（内存）传输的最大字节数 |
| 24 | maxTransferCountOnMessageInMemory | 32 | 单次Pull消息（内存）传输的最大条数 |
| 25 | maxTransferBytesOnMessageInDisk | 65536 | 单次Pull消息（磁盘）传输的最大字节数 |
| 26 | maxTransferCountOnMessageInDisk | 8 | 单次Pull消息（磁盘）传输的最大条数 |
| 27 | messageIndexEnable | true | 是否开启消息索引功能 |
| 28 | messageIndexSafe | false | 是否提供安全的消息索引机制，索引保证不丢 |
| 29 | haMasterAddress |  | 在Slave上直接设置Master地址，默认从Name Server上自动获取，也可以手工强制配置 |
| 30 | cleanFileForciblyEnable | true | 磁盘满、且无过期文件情况下 TRUE 表示强制删除文件，优先保证服务可用 FALSE 标记服务不可用，文件不删除 |
| 31 | enablePropertyFilter | true | 是否开启过滤查询 |

**Consumer参数**

|  |  |  |  |
| --- | --- | --- | --- |
| DefaultMQProducer、TransactionMQProducer、DefaultMQPushConsumer、DefaultMQPullConsumer都继承与ClientConfig类，  ClientConfig为客户端的公共配置类。客户端的配置都是get、set形式，每个参数都可以用spring来配置，也可以在代码中配置，  例如namesrvAddr这个参数可以这样配置，其他参数同理。  producer.setNamesrvAddr("192.168.0.1:9876"); | | | |
| **■客户端的公共配置** |  |  |  |
| **#** | **参数名** | **默认值** | **说明** |
| 1 | namesrvAddr |  | Name Server地址列表，多个NameServer地址用分号隔开 |
| 2 | clientIP | 本机IP | 客户端本机IP地址，某些机器会发生无法识别客户端IP地址情况，需要应用在代码中强制指定 |
| 3 | instanceName | DEFAULT | 客户端实例名称，客户端创建的多个Producer、Consumer实际是共用一个内部实例（这个实例包含网络连接、线程资源等） |
| 4 | clientCallbackExecutorThreads | 4 | 通信层异步回调线程数 |
| 5 | pollNameServerInteval | 30000 | 轮询Name Server间隔时间，单位毫秒 |
| 6 | heartbeatBrokerInterval | 30000 | 向Broker发送心跳间隔时间，单位毫秒 |
| 7 | persistConsumerOffsetInterval | 5000 | 持久化Consumer消费进度间隔时间，单位毫秒 |

**Producer参数**

|  |  |  |  |
| --- | --- | --- | --- |
| **Producer配置** | | | |
| **#** | **参数名** | **默认值** | **说明** |
| 1 | producerGroup | DEFAULT\_PRODUCER | Producer组名，多个Producer如果属于一个应用，发送同样的消息，则应该将它们归为同一组 |
| 2 | createTopicKey | TBW102 | 在发送消息时，自动创建服务器不存在的topic，需要指定Key。 |
| 3 | defaultTopicQueueNums | 4 | 在发送消息时，自动创建服务器不存在的topic，默认创建的队列数 |
| 4 | sendMsgTimeout | 10000 | 发送消息超时时间，单位毫秒 |
| 5 | compressMsgBodyOverHowmuch | 4096 | 消息Body超过多大开始压缩（Consumer收到消息会自动解压缩），单位字节 |
| 6 | retryAnotherBrokerWhenNotStoreOK | FALSE | 如果发送消息返回sendResult，但是sendStatus!=SEND\_OK，是否重试发送 |
| 7 | maxMessageSize | 131072 | 客户端限制的消息大小，超过报错，同时服务端也会限制 |
| 8 | transactionCheckListener |  | 事务消息回查监听器，如果发送事务消息，必须设置 |
| 9 | checkThreadPoolMinSize | 1 | Broker回查Producer事务状态时，线程池大小 |
| 10 | checkThreadPoolMaxSize |  | Broker回查Producer事务状态时，线程池大小 |
| 11 | checkRequestHoldMax | 2000 | Broker回查Producer事务状态时，Producer本地缓冲请求队列大小 |
|  |  |  |  |
| **Push Consumer配置** | | | |
| **#** | **参数名** | **默认值** | **说明** |
| 1 | consumerGroup | DEFAULT\_CONSUMER | Consumer组名，多个Consumer如果属于一个应用，订阅同样的消息，且消费逻辑一致，则应该将它们归为同一组 |
| 2 | messageModel | CLUSTERING | 消息模型，支持以下两种  1、集群消费(CLSUTER)  2、广播消费(BROADCASTING) |
| 3 | consumeFromWhere | CONSUME\_FROM\_LAST\_OFFSET | Consumer启动后，默认从什么位置开始消费  1、CONSUME\_FROM\_LAST\_OFFSET：默认策略，从该队列最尾开始消费，即跳过历史消息  2、CONSUME\_FROM\_FIRST\_OFFSET：从队列最开始开始消费，即历史消息（还储存在broker的）全部消费一遍  3、CONSUME\_FROM\_TIMESTAMP：从某个时间点开始消费，和setConsumeTimestamp()配合使用，默认是半个小时以前 |
| 4 | allocateMessageQueueStrategy | AllocateMessageQueueAveragely | Rebalance算法实现策略 |
| 5 | subscription | {} | 订阅关系 |
| 6 | messageListener |  | 消息监听器 |
| 7 | offsetStore |  | 消费进度存储 |
| 8 | consumeThreadMin | 10 | 消费线程池数量 |
| 9 | consumeThreadMax | 20 | 消费线程池数量 |
| 10 | consumeConcurrentlyMaxSpan | 2000 | 单队列并行消费允许的最大跨度 |
| 11 | pullThresholdForQueue | 1000 | 拉消息本地队列缓存消息最大数 |
| 12 | pullInterval | 0 | 拉消息间隔，由于是长轮询，所以为0，但是如果应用为了流控，也可以设置大于0的值，单位毫秒 |
| 13 | consumeMessageBatchMaxSize | 1 | 批量消费，一次消费多少条消息 |
| 14 | pullBatchSize | 32 | 批量拉消息，一次最多拉多少条 |
|  |  |  |  |
| **Pull Consumer配置** | | | |
| # | 参数名 | 默认值 | 说明 |
| 1 | consumerGroup | DEFAULT\_CONSUMER | Consumer组名，多个Consumer如果属于一个应用，订阅同样的消息，且消费逻辑一致，则应该将它们归为同一组 |
| 2 | brokerSuspendMaxTimeMillis | 20000 | 长轮询，Consumer拉消息请求在Broker挂起最长时间，单位毫秒 |
| 3 | consumerTimeoutMillisWhenSuspend | 30000 | 长轮询，Consumer拉消息请求在Broker挂起超过指定时间，客户端认为超时，单位毫秒 |
| 4 | consumerPullTimeoutMillis | 10000 | 非长轮询，拉消息超时时间，单位毫秒 |
| 5 | messageModel | BROADCASTING | 消息模型，支持以下两种  1、集群消费  2、广播消费 |
| 6 | messageQueueListener |  | 监听队列变化 |
| 7 | offsetStore |  | 消费进度存储 |
| 8 | registerTopics | [] | 注册的topic集合 |
| 9 | allocateMessageQueueStrategy | AllocateMessageQueueAveragely | Rebalance算法实现策略 |

**Meesage数据结构**

|  |  |  |  |
| --- | --- | --- | --- |
| Message数据结构各个字段都可以通过get、set方式访问，例如访问topic：  msg.getTopic();  msg.setTopic("test"); | | | |
| 字段名 | 默认值 | 必填 | 说明 |
| Topic | null | ○ | 线下环境不需要申请，线上环境需要申请后才能使用 |
| Body | null | ○ | 二进制形式，序列化由应用决定，Producer与Consumer要协商好序列化形式。 |
| Tags | null |  | 类似于Gmail为每封邮件设置的标签，方便服务器过滤使用。目前只支持每个消息设置一个tag，所以也可以类比为Notify的MessageType概念。 |
| Keys | null |  | 代表这条消息的业务关键词，服务器会根据keys创建哈希索引，设置后，可以再Console系统根据Topic、Keys来查询消息，由于是哈希索引，请尽可能保证key唯一，例如订单号，商品ID等。 |
| Flag | 0 |  | 完全由应用来设置，RocketMQ不做敢于。 |
| DelayTimeLevel | 0 |  | 消息延时级别，0表示不延时，大于0会延时特定的时间才会被消费。 |
| WaitStoreMsgOK | TRUE |  | 表示消息是否在服务器罗盘后才返回应答。 |