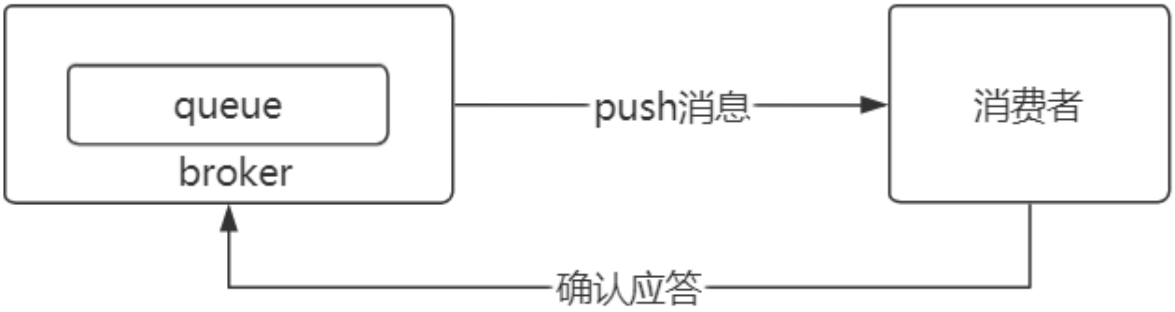
RocketMQ Push与Pull模式

RocketMQ在消费者层面拥有两种数据获取方式：

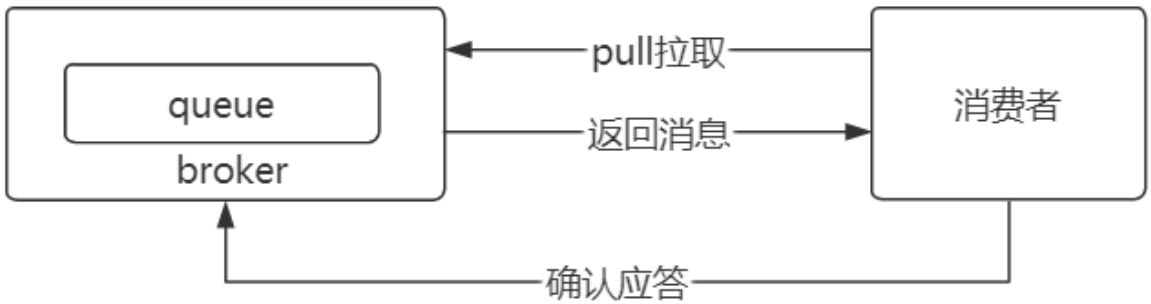
- 推送模式Push



推送模式顾名思义，是由Broker作为发起方，主动向消费者将最新产生的数据推送过来，Push模式使用的消费者类为DefaultMQPushConsumer，在前面我们一直在利用DefaultMQPushConsumer进行开发。

```
1 DefaultMQPushConsumer consumer = new DefaultMQPushConsumer("consumer-group");
```

- 拉取模式Pull



拉取模式是指由消费者作为发起方，定时向Broker发起队列查询请求，Broker将最近还没有消费的消息返回给消费者进行处理，拉取模式最显著的特点是发起者为消费方，定时拉取最近未消费数据，Broker返回后再有消费者确认接收应答。

Pull模式

在RocketMQ 4.6版本后，默认Pull模式不再使用DefaultMQPullConsumer拉取消息，改为使用DefaultLitePullConsumer，DefaultLitePullConsumer是更加轻量级的Pull模式消费者实现，以前使用DefaultMQPullConsumer实现Pull模式时要关注大量拉取时的开发细节，但这些在引入DefaultLitePullConsumer以后变的非常简单，RocketMQ做好了大量封装，让开发者只是用简单的几个API便可以实现复杂的拉取操作。

下面我们来看一下代码部分

生产者PullProducer

生产者部分和标准代码相同，这里我们模拟发送100条税务数据

```
1 DefaultMQProducer producer = new DefaultMQProducer("pg1");
```

```
2 producer.setNamesrvAddr("192.168.31.103:9876");
```

```
try {
    producer.start();
    for(int i = 1 ; i<= 100 ; i++) {
        String data = "id=" + i + "税务数据";
        Message message = new Message("tax-data", data.getBytes());
        SendResult result = producer.send(message);
        System.out.println("消息已发送：MsgId:" + result.getMsgId() + "，发送状态:" + resu
        Thread.sleep(1000);
    }
} catch (MQClientException | RemotingException | MQBrokerException | InterruptedExceptio
    e.printStackTrace();
} finally {
    try {
        producer.shutdown();
        System.out.println("链接已关闭");
    }catch (Exception e){
        e.printStackTrace();
    }
}
```

消费者PullConsumer

消费者部分变化较大，体现在三方面：

- 采用DefaultLitePullConsumer 对象拉取
- 调用poll方法拉取未消费数据
- 需要自定义获取间隔

```
//采用DefaultLitePullConsumer 对象拉取消息
DefaultLitePullConsumer consumer = new DefaultLitePullConsumer("consumer-group");
//注册NameServer地址
consumer.setNamesrvAddr("192.168.31.103:9876");
//订阅主题
consumer.subscribe("tax-data", "*");
// 启动消费者
consumer.start();
log.info("消费者启动成功，正在监听新消息");
int i = 0;
while (true) {
    ++i;
    //调用poll方法拉取未消费数据
    List<MessageExt> list = consumer.poll();
```

```
    int j = 0;
    if (list != null && list.size() > 0) {
        for (MessageExt ext : list) {
            j++;
            log.info("{}-{},{},{},{}", String.valueOf(i),String.valueOf(j), "Queue-" + ex
        }
    }
}
```

运行结果

- **情况1：先启动消费者，再启动生产者**

通过观察可以得到，消费者轮询每个队列，每次只取1个消息进行处理，因为发送消息的时候是1条1条发送的。

```
14:51:13.689 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 消费者启动成功，正在监听
14:51:30.539 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 4-1,Queue-3,3425,id=1秒
14:51:30.542 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 5-1,Queue-0,3639,id=2秒
14:51:30.544 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 6-1,Queue-1,3452,id=3秒
14:51:30.547 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 7-1,Queue-2,3438,id=4秒
14:51:30.550 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 8-1,Queue-3,3426,id=5秒
14:51:30.553 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 9-1,Queue-0,3640,id=6秒
14:51:30.556 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 10-1,Queue-1,3453,id=7
14:51:30.559 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 11-1,Queue-2,3439,id=8
14:51:30.562 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 12-1,Queue-3,3427,id=9
14:51:30.564 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 13-1,Queue-0,3641,id=10
14:51:30.568 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 14-1,Queue-1,3454,id=11
...
```

- **情况2：先启动生产者，在启动消费者**

通过观察得到，消费者在启动时会自动将之前积压在队列中的数据批量消费，默认每次10条，直到将所有积压数据消费掉然后再1条1条获取。类似情况下，如果生产者后续一次批量发送10条，消费者也是poll一次10条，最多不超过PullBatchSize默认10。

```
14:55:03.757 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 消费者启动成功，正在监听
14:55:03.990 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-1,Queue-1,3477,id=1秒
14:55:03.998 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-2,Queue-1,3478,id=5秒
14:55:03.998 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-3,Queue-1,3479,id=9秒
...
14:55:03.998 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-10,Queue-1,3486,id=3
```

```
14:55:03.998 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-1,Queue-0,3664,id=4秒
...
14:55:03.998 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-10,Queue-0,3673,id=4(
14:55:03.999 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 3-1,Queue-2,3463,id=2秒
...
14:55:04.006 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 12-5,Queue-1,3501,id=9
```

FAQ

poll方法默认自动提交，可以手动提交吗?

可以，需要两点调整：

- consumer.setAutoCommit(false); //关闭自动提交
- consumer.commitSync();//手动提交消费进度

注意：一定要在业务处理完毕后确保手动执行commitSync方法，否则下次消息会被重复拉取。

工作中更推荐使用这种方式，在处理业务后手动提交，可以避免数据因为业务失败导致的消息丢失。

完整代码如下：

```java
@Slf4j
public class PullConsumer {
    public static void main(String[] args) throws Exception {
        DefaultLitePullConsumer consumer = new DefaultLitePullConsumer("consumer-group")
        consumer.setNamesrvAddr("192.168.31.103:9876");
        consumer.subscribe("tax-data", "*");
        consumer.setAutoCommit(false); //关闭自动提交
        // 启动消费者
        consumer.start();
        log.info("消费者启动成功，正在监听新消息");
        int i = 0;
        while (true) {
            ++i;
            List<MessageExt> list = consumer.poll();
            int j = 0;
            if (list != null && list.size() > 0) {
                for (MessageExt ext : list) {
                    j++;
                    log.info("{}-{},{},{},{}", String.valueOf(i),String.valueOf(j), "Que
                }
            }
            consumer.commitSync(); //手动提交消费进度
        }
```

```
    }
}
```

如何调整批量获取数量?

```
consumer.setPullBatchSize(5);
```

运行结果

```
15:03:04.160 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 消费者启动成功，正在监听
15:03:04.389 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-1,Queue-0,3689,id=20
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-2,Queue-0,3690,id=21
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-3,Queue-0,3691,id=22
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-4,Queue-0,3692,id=23
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 1-5,Queue-0,3693,id=24
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-1,Queue-1,3502,id=30
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-2,Queue-1,3503,id=31
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-3,Queue-1,3504,id=32
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-4,Queue-1,3505,id=33
15:03:04.396 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 2-5,Queue-1,3506,id=34
15:03:04.397 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 3-1,Queue-3,3475,id=10
15:03:04.397 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 3-2,Queue-3,3476,id=11
15:03:04.397 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 3-3,Queue-3,3477,id=12
15:03:04.397 [main] INFO com.itlaoqi.rocketmq.pull.PullConsumer - 3-4,Queue-3,3478,id=13
```