Java应用接入RocketMQ消费数据

消费者接入过程

1. 新建Maven工程，添加依赖，注意Maven工程依赖与RocketMQ版本保持一致

```
<repositories>
    <repository>
        <id>aliyun</id>
        <name>aliyun</name>
        <url>https://maven.aliyun.com/repository/public</url>
    </repository>
</repositories>
<pluginRepositories>
    <pluginRepository>
        <id>aliyun</id>
        <name>aliyun</name>
        <url>https://maven.aliyun.com/repository/public</url>
    </pluginRepository>
</pluginRepositories>
...
<dependency>
    <groupId>org.apache.rocketmq</groupId>
    <artifactId>rocketmq-client</artifactId>
    <version>4.9.2</version>
</dependency>
```

2. 编写消费者接入代码

```
//创建消费者对象
DefaultMQPushConsumer consumer = new DefaultMQPushConsumer("cg1");
try {
    //设置NameServer节点
    consumer.setNamesrvAddr("192.168.31.103:9876");

    /*订阅主题，
    consumer.subscribe包含两个参数：
    topic: 说明消费者从Broker订阅哪一个主题，这一项要与Provider保持一致。
    subExpression: 子表达式用于筛选tags。
        同一个主题下可以包含很多不同的tags，subExpression用于筛选符合条件的tags进行接收。
        例如：设置为*，则代表接收所有tags数据。
        例如：设置为2020S4，则Broker中只有tags=2020S4的消息会被接收，而2020S2就会被排除在外。
    */
    consumer.subscribe("tax-data", "*");
    //创建监听，当有新的消息监听程序会及时捕捉并加以处理。
```

```
    consumer.registerMessageListener(new MessageListenerConcurrently() {
        public ConsumeConcurrentlyStatus consumeMessage(
                List<MessageExt> msgs, ConsumeConcurrentlyContext context) {
            //批量数据处理
            for (MessageExt msg : msgs) {
                System.out.println("消费者获取数据:" + msg.getMsgId() + "==>" + new
String(msg.getBody()));
            }
            //返回数据已接收标识
            return ConsumeConcurrentlyStatus.CONSUME_SUCCESS;
        }
    });
    //启动消费者，与Broker建立长连接，开始监听。
    consumer.start();
} catch (Exception e) {
    e.printStackTrace();
}
```