下载本模块安装包(kafka_client.zip)并解压后，请妥善放置其解压文件，安装完成后再次移动、改动或删除其解压文件将导致该模块无法正常使用。

本模块安装时应在命令行中转到安装包“setup.py“文件所在路径下，激活待安装的虚拟环境后执行如下代码：

```
python setup.py develop
```

另：本模块依赖于第三方类库confluent-kafka，该类库推荐python3版本3.8及以下。如果安装包安装失败，请尝试运行以下代码后重试：

```
pip install confluent-kafka
```

配置以及使用示例位于*kafka_client_usage.zip*压缩包, 请解压后运行

使用者只需关注生产者类、消费者类及*topic*操作中的“创建*topic*”方法

topic操作：

```
class CommunicationInitial(CommunicationInitialInterface)
```

查询topic列表

```
@staticmethod
def topicQuery(communicationCharactor: QueryInterface)
```

```
:param communicationCharactor: a instance of interface "QueryInterface"

:return: a list of topic name
```

可能异常：

```
TopicQueryFailed # 通常由于网络问题导致，发生后建议检查同Kafka服务器的网络联通性，或联系管理员检查Kafka服务健康度
```

创建topic

```
@staticmethod
def topicCreate(topic, confPath, num_partitions=1, replication_factor=1)
```

```
this method will retry for "retryLimit" (a param which is set in config) times
until success

:param topic: this param is the name of the topic that you want to create. type:
str

:param confPath: broker configuration, "bootstrap.servers" must be set

:param num_partitions: this param is the partition number of the topic that you
want to create. default: 1, type: int

:param replication_factor: this param is the replication number of the topic
that you want to create.
default: 1, type: int

:return: a dict of futures for each topic, keyed by the topic name. type:
dict(<topic_name, future>)
```

可能异常：

```
NoConfigFileException # 配置文件不存在
WrongConfigContextException #  配置文件内容有误
TopicCreateFailed # topic创建失败，发生后建议检查同Kafka服务器的网络联通性，或联系管理员检查Kafka服务健康度
```

删除topic

```
@staticmethod
def topicDelete(topic, confPath):
```

```
:param topic: this param is the name of the topic that you want to delete. type:
str

:param confPath: broker configuration, "bootstrap.servers" must be set

:return: a dict of futures for each topic, keyed by the topic name. type:
dict(<topic_name, future>)
```

可能异常:

```
NoConfigFileException # 配置文件不存在
WrongConfigContextException #  配置文件内容有误
TopicCreateFailed # topic删除失败
```

生产者：

```
class CommunicationProducer(CommunicationProducerInterface, QueryInterface)
```

调用者只需要关注***init***、***send***和***close***的用法

初始化

```
def __init__(self, confPath)
```

```
:param confPath: this param is the path of the producer config file that you
want to use. type: str
```

发送消息

```
send(self, topic: str, value: bytes, timeout: float = 1, key=None) -> None
```

```
:param topic: this param is the topic name that you want to send message to,
type: str

:param value: this param is the message context, type: bytes

:param timeout: this param is the timeout for sending a msg, but that doesn't
mean the msg sending is failed

:param key: this param isn't used currently

:return: None
```

可能异常：

```
WrongMessageValueType #  消息类型非bytes
```

关闭

```
def close(self, timeout: float = 1) -> None
```

```
release resources

:param: timeout: this param is the flush timeout before closed

:return: None
```

关闭动作：

推送生产者队列中的全部消息

消费者：

```
class CommunicationConsumer(CommunicationConsumerInterface, QueryInterface)
```

调用者只需要关注***init***、***receive***、□□□***timeStampReceive***和***stop***的用法

初始化

```
def __init__(self, confPath, consumerId)
```

```
:param confPath: this param is the path of the producer config file that you
want to use. type: str

:param consumerId: this param should be unique in the system, UUID suggested.
type: str
```

订阅

```
def subscribe(self, topic: str) -> None
```

```
:param topic: this param is the topic this consumer need to subscribe
```

可能异常：

```
TopicNotAvailableException #  topic不存在或其他原因订阅失败，发生后建议手动创建topic，或检查同Kafka服务器的网络联通性，或联系管理员检查Kafka服务健康度
```

取消订阅

```
def unsubscribe(self) -> None
```

接收消息

```
def receive(self) -> bytes
```

```
:return: unpacking message received in timeout. type: bytes or None(when there
is no message in timeout)
```

接收带有时间戳的消息

```
def timeStampReceive(self) -> list
```

```
:return: a list of timestamp and message value.
the first element in the list is a tulpe of timestamp. the first element is
timestamp type,
which is a number in 0, 1 or 2.
0 means the timestamp isn't available, in this case, the return timestamp should
be ignore.
1 means the return timestamp is the number of milliseconds of the message
creation time.
2 means the return timestamp is the number of milliseconds of the broker receive
time.
the first element in the list is bytes of  message value.
type: [(int, int), bytes] or None(when there is no message in timeout)
```

关闭

```
def close(self) -> None
```

关闭动作：

取消订阅、关闭消费者

配置文件说明

CommunicationInitial类的配置文件

配置文件路径：./communicationModuleUsage/config/initial-config.json

话题管理配置文件示例

```
{
“bootstrap.servers”: “server:60000”,
“retryLimit”: “3”
}
```

字段含义：

- **bootstrap.servers**：type：string。（下同）通信服务器地址，如Kafka服务部署方式或服务器IP有变，请询问管理员具体地址及使用方式。（在默认部署方式下，需在计算机中添加“<host名><IP地址>”映射，然后在此处填入host名和端口号。host名（或IP）与端口号之间以英文冒号分隔。）
- **retryLimit**： type：string，内容需为大于等于0的int型数字。`CommunicationInitial.topicCreate`方法重试次数。如果网络环境较差，建议将该字段设为较大值，但如果该字段值过大，将导致用户无法及时发现网络异常。

CommunicationProducer类的配置文件

配置文件路径：./communicationModuleUsage/config/producer-config.json

生产者配置文件示例

```
{
"bootstrap.servers": "server:60000",
"debugLogEnable": "false",
"linger.ms": "0",
"retryLimit": "0",
"realtimeFlush": "true”
}
```

字段含义：

- **debugLogEnable**： type：string，enum：{false，true}，大小写不敏感。（下同）是否开启生产者/消费者debug级别的日志记录，如果开启，则生产者/消费者将在每次生产/消费动作完成后记录一条debug级别的日志，在大量、低间隔进行生产/消费活动时建议关闭该日志。
- **linger.ms**：type：string，内容需为int型数字。对生产者发送速度有较高要求时建议将该字段设为较小值。

- **retryLimit**： type：string，内容需为大于等于0的int型数字。`CommunicationProducer.listTopics`方法重试次数。如果网络环境较差，建议将该字段设为较大值，但如果该字段值过大，将导致用户无法及时发现网络异常。
- **realtimeFlush**： type：string，enum：{false，true}，大小写不敏感。是否开启实时推送。如果开启该功能可以保证消息尽快发出，但开启后会导致`CommunicationProducer.send`方法在被调用后产生大致10ms的阻塞。

CommunicationConsumer类的配置文件

配置文件路径：./communicationModuleUsage/config/consumer-config.json

消费者配置文件示例

```
{
"bootstrap.servers": "server:60000",
"auto.offset.reset": "latest",
"enable.auto.commit": "False",
"retryLimit": "3",
"debugLogEnable": "False”
}
```

字段含义：

- **auto.offset.reset**： type：string，enum：{latest，earliest，none}，大小写敏感。消费者从消息队列的何处开始消费。该参数不建议改动，默认给出的设置将使消费者从其启动时的最新消息开始消费。
- **enable.auto.commit** ：该参数不建议改动。
- **retryLimit**： type：string，内容需为大于等于0的int型数字。`CommunicationConsumer.listTopics`方法重试次数。如果网络环境较差，建议将该字段设为较大值，但如果该字段值过大，将导致用户无法及时发现网络异常。