

```

println(s"fromOffsetMap-----==>${fromOffsetMap.size}")
val size = fromOffsetMap.size // 读取到的Offset大小
// 计算Offset存储记录的大小, 如果有记录则从记录处继续消费Offset, 如果没有记录则从Offset当前位置最大处开始消费
val inputDS : InputDStream[(String, String)] = if (size > 0){
    val messageHandler = (mmd: MessageAndMetadata[String,String]) => (mmd.key,
mmd.message)
    KafkaUtils.createDirectStream[String, String, StringDecoder, StringDecoder,
(String,String)](
        ssc, ParamsUtils.kafka.KAFKA_PARAMS, fromOffsetMap, messageHandler)
    } else {
    KafkaUtils.createDirectStream[String, String, StringDecoder, StringDecoder](
        ssc, ParamsUtils.kafka.KAFKA_PARAMS, topics.split(",").toSet)
    }
inputDS
}
}

```

上段逻辑从Zookeeper中获取Offset位置的方法
KafkaParaUtils.readOffSet的实现逻辑如下:

```

// 读取Kafka的Offset偏移量
def readOffSet(groupId:String, topic:String): Map[TopicAndPartition, Long] = {
    println("-----> 读取偏移量")
    val zkClient = getZKClient
    // Kafka分区
    val OffsetMap = collection.mutable.Map.empty[TopicAndPartition, Long]
    try {
        val partitionSeq = KafkaFunction.kafkaPartitionByTopic(zkClient, topic)
        partitionSeq.map { p =>
            // 默认路径
            //      println(s"zKGroupTopicDirs:
            ${KafkaFunction.zKGroupTopicDirs(groupId,topic)}\t;") //zKGroupTopicDirs
            val offsetPath = offsetPathFun(topic, groupId, p)
            println(s"offsetPath: ${offsetPath}")

            // 偏移量查询结果
            val offsetTP = KafkaFunction.offsetStatTuple(zkClient,offsetPath)
            println(s"offsetTP:${offsetTP}")

            if (offsetTP != null) {
                OffsetMap.put(TopicAndPartition(topic, p), offsetTP._1.toLong)
            }
        }
    } finally {
        zkClient.close()
    }
    OffsetMap.toMap
}

```

对应地, 将消费后当前的Offset偏移量存储到Zookeeper中, 实现逻辑如下:
