通过编程方式获取Kafka中Topic的Metadata信息

如果我们需要通过编程的方式来获取到Kafka中某个Topic的所有分区、副本、每个分区的Leader（所在机器及其端口等信息），所有分区副本所在机器的信息和ISR机器的信息等（特别是在使用Kafka的Simple
API来编写SimpleConsumer的情况）。这一切可以通过发送TopicMetadataRequest请求到Kafka Server中获取。代码片段如下所示：

```
def findLeader(topic: String): Unit = {
  val consumer = connect("www.iteblog.com", 9092)
  val req = TopicMetadataRequest(TopicMetadataRequest.CurrentVersion,
       0, kafkaGroupId, List(topic))

  val topicMetadataResponse = consumer.send(req)
  val topicsMetadataSet = topicMetadataResponse.topicsMetadata

  topicsMetadataSet.foreach { topicMetadata =>
    println(topicMetadata.topic)
    val metadataSet = topicMetadata.partitionsMetadata

    metadataSet.foreach { metadata =>
      val partitionId = metadata.partitionId
      val isr = metadata.isr.map(_.connectionString).mkString("[", ",", "]")
      val replicas = metadata.replicas.map(_.connectionString).mkString("[", ",", "]")
      val leader = metadata.leader.map (_.connectionString).get
      println(s"₩tPartition: $partitionId, Leader: $leader Replicas: $replicas ISR: $isr")
    }
  }
}
```

TopicMetadataRequest是一个case class，其各个参数如下：

```
case class TopicMetadataRequest(val versionId: Short,
                                val correlationId: Int,
                                val clientId: String,
                                val topics: Seq[String])
```

构造完成TopicMetadataRequest之后，通过SimpleConsumer的send方法发送请求，然后返回TopicMetadataResponse对象，其中就包含了Topic各个分区的相关信息，我们运行这个函数，可以得到以下的信息：

```
iteblog
Partition: 0, Leader: www.iteblog.com:9091 Replicas: [www.iteblog.com:9091] ISR: [www.iteblog.com:9091]
Partition: 1, Leader: www.iteblog.com:9097 Replicas: [www.iteblog.com:9097] ISR: [www.iteblog.com:9097]
Partition: 2, Leader: www.iteblog.com:9095 Replicas: [www.iteblog.com:9095] ISR: [www.iteblog.com:9095]
Partition: 3, Leader: www.iteblog.com:9096 Replicas: [www.iteblog.com:9096] ISR: [www.iteblog.com:9096]
Partition: 4, Leader: www.iteblog.com:9094 Replicas: [www.iteblog.com:9094] ISR: [www.iteblog.com:9094]
Partition: 5, Leader: www.iteblog.com:9092 Replicas: [www.iteblog.com:9092] ISR: [www.iteblog.com:9092]
Partition: 6, Leader: www.iteblog.com:9093 Replicas: [www.iteblog.com:9093] ISR: [www.iteblog.com:9093]
```

这个输出是不是很熟悉，是的，输出的结果类似于运行以下的Kafka自带系统命令：

```
[iteblog@www.iteblog.com kafka]$ ./bin/kafka-topics.sh --topic iteblog --describe  ₩
   --zookeeper www.iteblog.com
Topic:iteblog PartitionCount:7  ReplicationFactor:2 Configs:
  Topic: iteblog  Partition: 0  Leader: 1 Replicas: 1 Isr: 1
  Topic: iteblog  Partition: 1  Leader: 7 Replicas: 7 Isr: 7
  Topic: iteblog  Partition: 2  Leader: 5 Replicas: 5 Isr: 5
  Topic: iteblog  Partition: 3  Leader: 6 Replicas: 6 Isr: 6
  Topic: iteblog  Partition: 4  Leader: 4 Replicas: 4 Isr: 4
  Topic: iteblog  Partition: 5  Leader: 2 Replicas: 2 Isr: 2
  Topic: iteblog  Partition: 6  Leader: 3 Replicas: 3 Isr: 3
```

如果我们设置空的topic的列表，如：TopicMetadataRequest(TopicMetadataRequest.CurrentVersion, 0, kafkaGroupId, Seq())，那么我们可以获取Kafka server中所有Topic的信息。