定长记录输入格式

FixedLengthBinaryInputFormat

+ recordLength : int

+ def isSplitable(context: JobContext, filename: Path): Boolean
+ def computeSplitSize(blockSize: Long, minSize: Long, maxSize: Long): Long
+ def createRecordReader(split: InputSplit, context: TaskAttemptContext) : RecordReader[LongWritable, BytesWritable]

FixedLengthBinaryInputFormat(Object)

+ def getRecordLength(context: JobContext): Int

FixedLengthBinaryRecordReader

+ splitStart: long
+ splitEnd:long
+ currentPosition:int
+ recordLength
+ fileInputStream:FSDataInputStream
+ recordKey:LongWritable
+ recordValue:BytesWritable

+ def close(): Unit
+ def getCurrentKey: LongWritable
+ def getCurrentValue: BytesWritable
+ def getProgress: Float
+ def initialize(inputSplit: InputSplit, context: TaskAttemptContext): Unit
+ def nextKeyValue(): Boolean

全文输入格式

WholeTextFileInputFormat

+ def isSplitable(context: JobContext, file: Path): Boolean
+ def createRecordReader(split: InputSplit, context: TaskAttemptContext): RecordReader[Text, Text]
+ def setMinPartitions(context: JobContext, minPartitions: Int): Unit

WholeTextFileRecordReader

+ split: CombineFileSplit
+ context: TaskAttemptContext
+ index: Integer
+ path:Path
+ fs:FileSystem
+ processed:Boolean
+ key:Text
+ value:Text

+ def initialize(split: InputSplit, context: TaskAttemptContext): Unit
+ def close(): Unit
+ def getProgress: Float
+ def getCurrentKey: Text
+ def getCurrentValue: Text
+ def nextKeyValue(): Boolean

流式输入形式

Container

StreamFileInputFormat

+ attribute1:type = defaultValue
+ attribute2:type
- attribute3:type

+ def isSplitable(context: JobContext, file: Path): Boolean = false
+ def createRecordReader(split: InputSplit, taContext: TaskAttemptContext): RecordReader[String, T]
+ def setMinPartitions(sc: SparkContext, context: JobContext, minPartitions: Int): Unit

StreamRecordReader

- split: CombineFileSplit
- context: TaskAttemptContext
- index: Integer

+ def parseStream(inStream: PortableDataStream): PortableDataStream

StreamInputFormat

+ def createRecordReader(split: InputSplit, taContext: TaskAttemptContext): CombineFileRecordReader[String, PortableDataStream]

StreamBasedRecordReader

- split: CombineFileSplit
- context: TaskAttemptContext
- index: Integer
+ processed:Boolean
+ key
+ value

+ def initialize(split: InputSplit, context: TaskAttemptContext): Unit
+ def close(): Unit
+ def getProgress: Float
+ def getCurrentKey: String
+ def getCurrentValue: T
+ def parseStream(inStream: PortableDataStream): T
+ def nextKeyValue: Boolean

PortableDataStream

- isplit: CombineFileSplit
- context: TaskAttemptContext
- index: Integer
+ confBytes : byte[]
+ splitBytes : byte[]
+ split: CombineFileSplit
+ conf: Configuration
+ path: String

+ def open(): DataInputStream
+ def toArray(): Array[Byte]
+ def getPath(): String
+ def getConfiguration: Configuration

Configurable

+ conf:Configuration

+ def setConf(c: Configuration): Unit
+ def getConf: Configuration

ConfigurableCombineFileRecordReader

- split: InputSplit
- context: TaskAttemptContext
- recordReaderClass: Class[_ <: RecordReader[K, V]

+ def initNextRecordReader(): Boolean
+ def setConf(c: Configuration): Unit