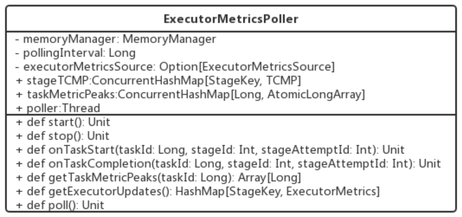
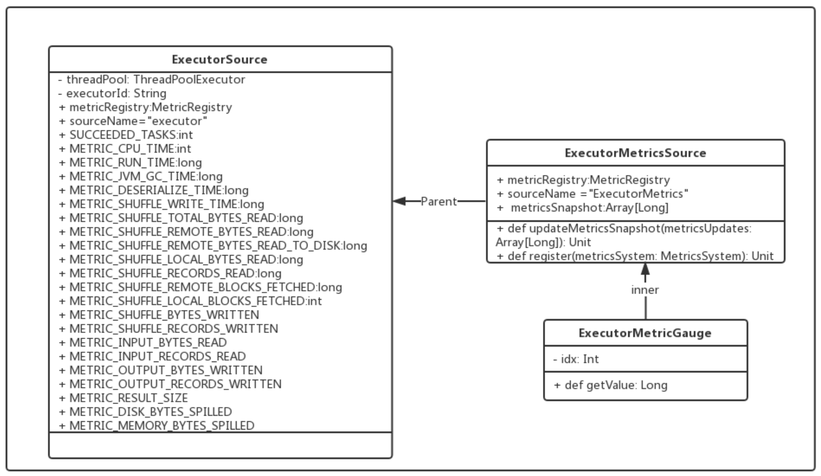
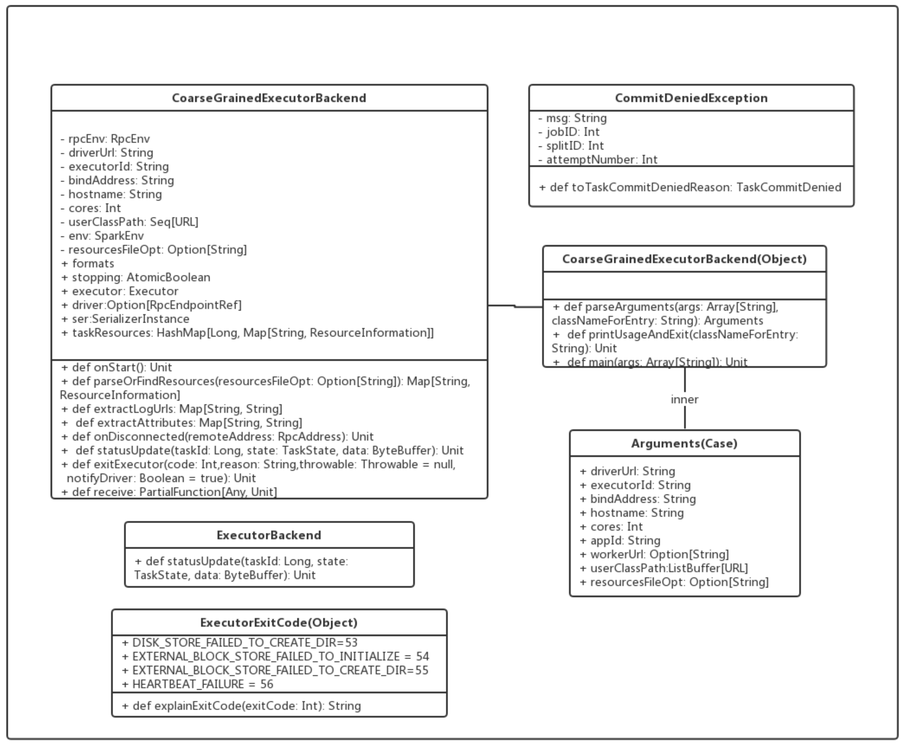
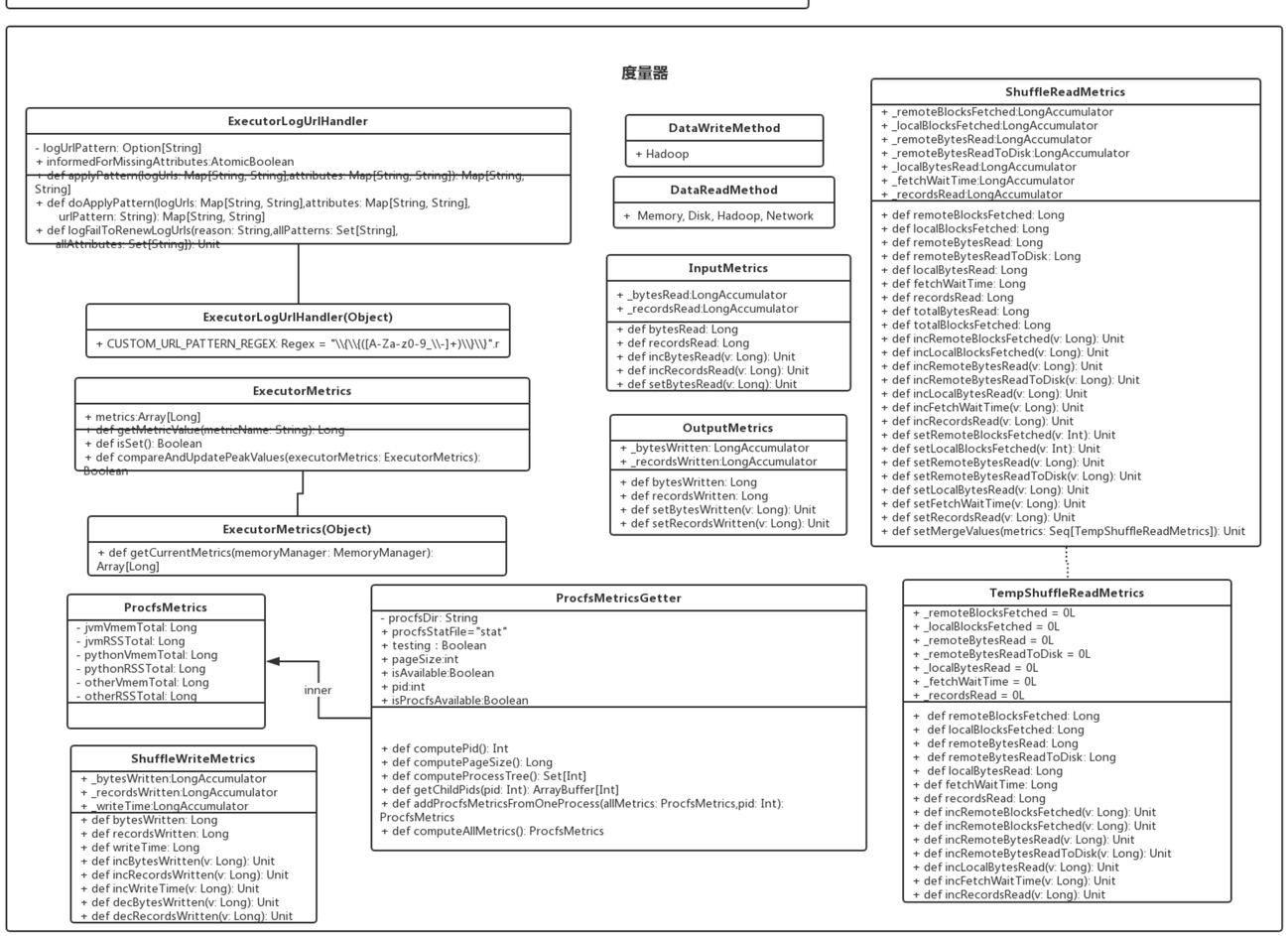
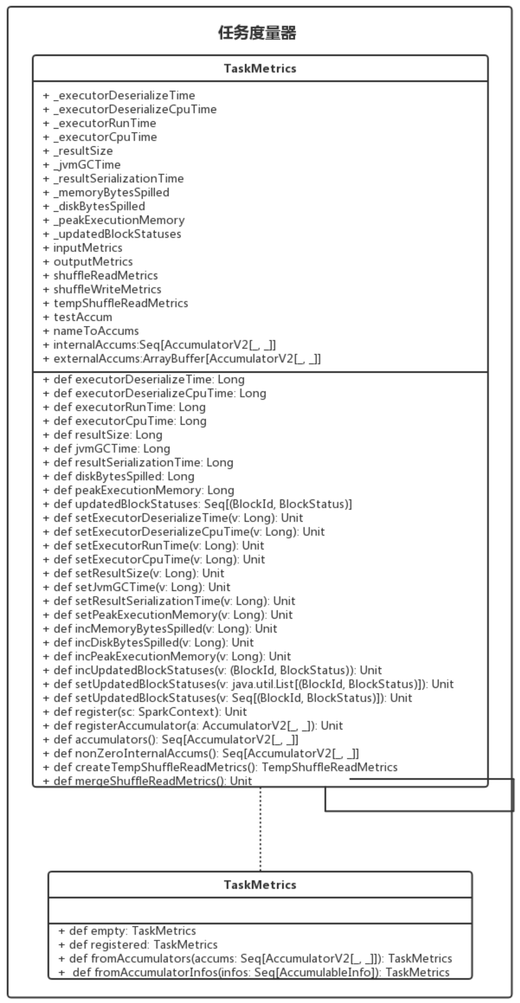

ExecutorMetricsPoller
- memoryManager: MemoryManager
- pollingInterval: Long
- executorMetricsSource: Option[ExecutorMetricsSource]
+ stageTCMP:ConcurrentHashMap[StageKey, TCMP]
+ taskMetricPeaks:ConcurrentHashMap[Long, AtomicLongArray]
+ poller:Thread
+ def start(): Unit
+ def stop(): Unit
+ def onTaskStart(taskId: Long, stageId: Int, stageAttemptId: Int): Unit
+ def onTaskCompletion(taskId: Long, stageId: Int, stageAttemptId: Int): Unit
+ def getTaskMetricPeaks(taskId: Long): Array[Long]
+ def getExecutorUpdates(): HashMap[StageKey, ExecutorMetrics]
+ def poll(): Unit


ExecutorSource
- threadPool: ThreadPoolExecutor
- executorId: String
+ metricRegistry:MetricRegistry
+ sourceName="executor"
+ SUCCEEDED_TASKS:int
+ METRIC_CPU_TIME:int
+ METRIC_RUN_TIME:long
+ METRIC_JVM_GC_TIME:long
+ METRIC_DESERIALIZE_TIME:long
+ METRIC_SHUFFLE_WRITE_TIME:long
+ METRIC_SHUFFLE_TOTAL_BYTES_READ:long
+ METRIC_SHUFFLE_REMOTE_BYTES_READ:long
+ METRIC_SHUFFLE_REMOTE_BYTES_READ_TO_DISK:long
+ METRIC_SHUFFLE_LOCAL_BYTES_READ:long
+ METRIC_SHUFFLE_RECORDS_READ:long
+ METRIC_SHUFFLE_REMOTE_BLOCKS_FETCHED:long
+ METRIC_SHUFFLE_LOCAL_BLOCKS_FETCHED:int
+ METRIC_SHUFFLE_BYTES_WRITTEN
+ METRIC_SHUFFLE_RECORDS_WRITTEN
+ METRIC_INPUT_BYTES_READ
+ METRIC_INPUT_RECORDS_READ
+ METRIC_OUTPUT_BYTES_WRITTEN
+ METRIC_OUTPUT_RECORDS_WRITTEN
+ METRIC_RESULT_SIZE
+ METRIC_DISK_BYTES_SPILLED
+ METRIC_MEMORY_BYTES_SPILLED
Parent
ExecutorMetricsSource
+ metricRegistry:MetricRegistry
+ sourceName = "ExecutorMetrics"
+ metricsSnapshot:Array[Long]
+ def updateMetricsSnapshot(metricsUpdates: Array[Long]): Unit
+ def register(metricsSystem: MetricsSystem): Unit
inner
ExecutorMetricGauge
- idx: Int
+ def getValue: Long


CoarseGrainedExecutorBackend
- rpcEnv: RpcEnv
- driverUrl: String
- executorId: String
- bindAddress: String
- hostname: String
- cores: Int
- userClassPath: Seq[URL]
- env: SparkEnv
- resourcesFileOpt: Option[String]
+ formats
+ stopping: AtomicBoolean
+ executor: Executor
+ driver:Option[RpcEndpointRef]
+ ser:SerializerInstance
+ taskResources: HashMap[Long, Map[String, ResourceInformation]]
+ def onStart(): Unit
+ def parseOrFindResources(resourcesFileOpt: Option[String]): Map[String, ResourceInformation]
+ def extractLogUrls: Map[String, String]
+ def extractAttributes: Map[String, String]
+ def onDisconnected(remoteAddress: RpcAddress): Unit
+ def statusUpdate(taskId: Long, state: TaskState, data: ByteBuffer): Unit
+ def exitExecutor(code: Int,reason: String,throwable: Throwable = null, notifyDriver: Boolean = true): Unit
+ def receive: PartialFunction[Any, Unit]
CommitDeniedException
- msg: String
- jobID: Int
- splitID: Int
- attemptNumber:: Int
+ def toTaskCommitDeniedReason: TaskCommitDenied
CoarseGrainedExecutorBackend(Object)
+ def parseArguments(args: Array[String], classNameForEntry: String): Arguments
+ def printUsageAndExit(classNameForEntry: String): Unit
+ def main(args: Array[String]): Unit
inner
Arguments(Case)
+ driverUrl: String
+ executorId: String
+ bindAddress: String
+ hostname: String
+ cores: Int
+ appId: String
+ workerUrl: Option[String]
+ userClassPath:ListBuffer[URL]
+ resourcesFileOpt: Option[String]
ExecutorBackend
+ def statusUpdate(taskId: Long, state: TaskState, data: ByteBuffer): Unit
ExecutorExitCode(Object)
+ DISK_STORE_FAILED_TO_CREATE_DIR=53
+ EXTERNAL_BLOCK_STORE_FAILED_TO_INITIALIZE = 54
+ EXTERNAL_BLOCK_STORE_FAILED_TO_CREATE_DIR=55
+ HEARTBEAT_FAILURE = 56
+ def explainExitCode(exitCode: Int): String


度量器
ExecutorLogUrlHandler
- logUrlPattern: Option[String]
+ informedForMissingAttributes:AtomicBoolean
+ def applyPattern(logUrls: Map[String, String],attributes: Map[String, String]): Map[String, String]
+ def doApplyPattern(logUrls: Map[String, String],attributes: Map[String, String], urlPattern: String): Map[String, String]
+ def logFailToRenewLogUrls(reason: String,allPatterns: Set[String], allAttributes: Set[String]): Unit
ExecutorLogUrlHandler(Object)
+ CUSTOM_URL_PATTERN_REGEX: Regex = "\\{\\{([A-Za-z0-9_\\-]+)\\}\\}".r
ExecutorMetrics
+ metrics:Array[Long]
+ def getMetricValue(metricName: String): Long
+ def isSet(): Boolean
+ def compareAndUpdatePeakValues(executorMetrics: ExecutorMetrics): Boolean
ExecutorMetrics(Object)
+ def getCurrentMetrics(memoryManager: MemoryManager): Array[Long]
ProcfsMetrics
- jvmVmemTotal: Long
- jvmRSSTotal: Long
- pythonVmemTotal: Long
- pythonRSSTotal: Long
- otherVmemTotal: Long
- otherRSSTotal: Long
inner
ProcfsMetricsGetter
- procfsDir: String
+ procfsStatFile="stat"
+ testing : Boolean
+ pageSize:int
+ isAvailable:Boolean
+ pid:int
+ isProcfsAvailable:Boolean
+ def computePid(): Int
+ def computePageSize(): Long
+ def computeProcessTree(): Set[Int]
+ def getChildPids(pid: Int): ArrayBuffer[Int]
+ def addProcfsMetricsFromOneProcess(allMetrics: ProcfsMetrics,pid: Int): ProcfsMetrics
+ def computeAllMetrics(): ProcfsMetrics
ShuffleWriteMetrics
+ _bytesWritten:LongAccumulator
+ _recordsWritten:LongAccumulator
+ _writeTime:LongAccumulator
+ def bytesWritten: Long
+ def recordsWritten: Long
+ def writeTime: Long
+ def incBytesWritten(v: Long): Unit
+ def incRecordsWritten(v: Long): Unit
+ def incWriteTime(v: Long): Unit
+ def decBytesWritten(v: Long): Unit
+ def decRecordsWritten(v: Long): Unit
DataWriteMethod
+ Hadoop
DataReadMethod
+ Memory, Disk, Hadoop, Network
InputMetrics
+ _bytesRead:LongAccumulator
+ _recordsRead:LongAccumulator
+ def bytesRead: Long
+ def recordsRead: Long
+ def incBytesRead(v: Long): Unit
+ def incRecordsRead(v: Long): Unit
+ def setBytesRead(v: Long): Unit
OutputMetrics
+ _bytesWritten: LongAccumulator
+ _recordsWritten:LongAccumulator
+ def bytesWritten: Long
+ def recordsWritten: Long
+ def setBytesWritten(v: Long): Unit
+ def setRecordsWritten(v: Long): Unit
ShuffleReadMetrics
+ _remoteBlocksFetched:LongAccumulator
+ _localBlocksFetched:LongAccumulator
+ _remoteBytesRead:LongAccumulator
+ _remoteBytesReadToDisk:LongAccumulator
+ _localBytesRead:LongAccumulator
+ _fetchWaitTime:LongAccumulator
+ _recordsRead:LongAccumulator
+ def remoteBlocksFetched: Long
+ def localBlocksFetched: Long
+ def remoteBytesRead: Long
+ def remoteBytesReadToDisk: Long
+ def localBytesRead: Long
+ def fetchWaitTime: Long
+ def recordsRead: Long
+ def totalBytesRead: Long
+ def totalBlocksFetched: Long
+ def incRemoteBlocksFetched(v: Long): Unit
+ def incLocalBlocksFetched(v: Long): Unit
+ def incRemoteBytesRead(v: Long): Unit
+ def incRemoteBytesReadToDisk(v: Long): Unit
+ def incLocalBytesRead(v: Long): Unit
+ def incFetchWaitTime(v: Long): Unit
+ def incRecordsRead(v: Long): Unit
+ def setRemoteBlocksFetched(v: Int): Unit
+ def setLocalBlocksFetched(v: Int): Unit
+ def setRemoteBytesRead(v: Long): Unit
+ def setRemoteBytesReadToDisk(v: Long): Unit
+ def setLocalBytesRead(v: Long): Unit
+ def setFetchWaitTime(v: Long): Unit
+ def setRecordsRead(v: Long): Unit
+ def setMergeValues(metrics: Seq[TempShuffleReadMetrics]): Unit
TempShuffleReadMetrics
+ _remoteBlocksFetched = 0L
+ _localBlocksFetched = 0L
+ _remoteBytesRead = 0L
+ _remoteBytesReadToDisk = 0L
+ _localBytesRead = 0L
+ _fetchWaitTime = 0L
+ _recordsRead = 0L
+ def remoteBlocksFetched: Long
+ def localBlocksFetched: Long
+ def remoteBytesRead: Long
+ def remoteBytesReadToDisk: Long
+ def localBytesRead: Long
+ def fetchWaitTime: Long
+ def recordsRead: Long
+ def incRemoteBlocksFetched(v: Long): Unit
+ def incRemoteBlocksFetched(v: Long): Unit
+ def incRemoteBytesRead(v: Long): Unit
+ def incRemoteBytesReadToDisk(v: Long): Unit
+ def incLocalBytesRead(v: Long): Unit
+ def incFetchWaitTime(v: Long): Unit
+ def incRecordsRead(v: Long): Unit


任务度量器
TaskMetrics
+ _executorDeserializeTime
+ _executorDeserializeCpuTime
+ _executorRunTime
+ _executorCpuTime
+ _resultSize
+ _jvmGCTime
+ _resultSerializationTime
+ _memoryBytesSpilled
+ _diskBytesSpilled
+ _peakExecutionMemory
+ _updatedBlockStatuses
+ inputMetrics
+ outputMetrics
+ shuffleReadMetrics
+ shuffleWriteMetrics
+ tempShuffleReadMetrics
+ testAccum
+ nameToAccums
+ internalAccums:Seq[AccumulatorV2[_, _]]
+ externalAccums:ArrayBuffer[AccumulatorV2[_, _]]
+ def executorDeserializeTime: Long
+ def executorDeserializeCpuTime: Long
+ def executorRunTime: Long
+ def executorCpuTime: Long
+ def resultSize: Long
+ def jvmGCTime: Long
+ def resultSerializationTime: Long
+ def diskBytesSpilled: Long
+ def peakExecutionMemory: Long
+ def updatedBlockStatuses: Seq[(BlockId, BlockStatus)]
+ def setExecutorDeserializeTime(v: Long): Unit
+ def setExecutorDeserializeCpuTime(v: Long): Unit
+ def setExecutorRunTime(v: Long): Unit
+ def setExecutorCpuTime(v: Long): Unit
+ def setResultSize(v: Long): Unit
+ def setJvmGCTime(v: Long): Unit
+ def setResultSerializationTime(v: Long): Unit
+ def setPeakExecutionMemory(v: Long): Unit
+ def incMemoryBytesSpilled(v: Long): Unit
+ def incDiskBytesSpilled(v: Long): Unit
+ def incPeakExecutionMemory(v: Long): Unit
+ def incUpdatedBlockStatuses(v: (BlockId, BlockStatus)): Unit
+ def setUpdatedBlockStatuses(v: java.util.List[(BlockId, BlockStatus)]): Unit
+ def setUpdatedBlockStatuses(v: Seq[(BlockId, BlockStatus)]): Unit
+ def register(sc: SparkContext): Unit
+ def registerAccumulator(a: AccumulatorV2[_, _]): Unit
+ def accumulators(): Seq[AccumulatorV2[_, _]]
+ def nonZeroInternalAccums(): Seq[AccumulatorV2[_, _]]
+ def createTempShuffleReadMetrics(): TempShuffleReadMetrics
+ def mergeShuffleReadMetrics(): Unit
TaskMetrics
+ def empty: TaskMetrics
+ def registered: TaskMetrics
+ def fromAccumulators(accums: Seq[AccumulatorV2[_, _]]): TaskMetrics
+ def fromAccumulatorInfos(infos: Seq[AccumulableInfo]): TaskMetrics


执行器
TaskRunner
- execBackend: ExecutorBackend
- taskDescription: TaskDescription
+ taskId:taskDescription.taskId
+ threadName:String
+ taskName:String
+ reasonIfKilled:Option[String]
+ threadId:int
+ finished:Boolean
+ startGCTime
+ task:Task[Any]
+ def hasFetchFailure: Boolean
+ def setTaskFinishedAndClearInterruptStatus(): Unit
+ def collectAccumulatorsAndResetStatusOnFailure(taskStartTimeNs: Long)(Seq(),Seq())
+ def run(): Unit
Inner
Executor
- executorId: String
- executorHostname: String
- env: SparkEnv
- userClassPath: Seq[URL]
- isLocal: Boolean
- uncaughtExceptionHandler: UncaughtExceptionHandler
+ executorShutdown:AtomicBoolean
+ currentFiles: HashMap[String, Long]
+ currentJars:HashMap[String, Long]
+ EMPTY_BYTE_BUFFER
+ conf:SparkConf
+ threadPool:ExecutorService
+ executorSource:ExecutorSource
+ taskReaperPool
+ taskReaperForTask:HashMap[Long, TaskReaper]
+ executorMetricsSource
+ userClassPathFirst
+ taskReaperEnabled
+ urlClassLoader
+ replClassLoader
+ plugins
+ maxDirectResultSize
+ maxResultSize
+ runningTasks:ConcurrentHashMap[Long, TaskRunner]
+ HEARTBEAT_MAX_FAILURES
+ HEARTBEAT_DROP_ZEROES
+ HEARTBEAT_INTERVAL_MS
+ METRICS_POLLING_INTERVAL_MS
+ pollOnHeartbeat:Boolean
+ metricsPoller:ExecutorMetricsPoller
+ heartbeater:Heartbeater
+ heartbeatReceiverRef
+ heartbeatFailures:int
+ def numRunningTasks: Int
+ def launchTask(context: ExecutorBackend, taskDescription: TaskDescription): Unit
+ def killTask(taskId: Long, interruptThread: Boolean, reason: String): Unit
+ def killAllTasks(interruptThread: Boolean, reason: String) : Unit
+ def stop(): Unit
+ def computeTotalGcTime(): Long
+ def createClassLoader(): MutableURLClassLoader
+ def addReplClassLoaderIfNeeded(parent: ClassLoader): ClassLoader
+ def updateDependencies(newFiles: Map[String, Long], newJars: Map[String, Long]): Unit
+ def reportHeartBeat(): Unit
Executor(Object)
+ taskDeserializationProps: ThreadLocal[Properties]
Inner
TaskReaper
- taskRunner:: TaskRunner
- interruptThread: Boolean
- reason: String
+ taskId
+ killPollingIntervalMs
+ killTimeoutNs
+ takeThreadDump
+ def run(): Unit