Log File Scavenger Hunt

Do you want to learn some key concepts that will make Druid log files more meaningful and helpful? This scavenger hunt will familiarise you with packages and terms that will help you navigate and grok Druid log files.

If you have not yet taken the Apache Druid® Logs course, you can find it [here](https://learn.imply.io/apache-druid-logs). The logs course will give you a foundational understanding, and skills you can use to become familiar with Druid log files.

Below, you’ll find some interesting search terms. Plug these into your preferred tool, whether that’s *grep* or *less* or something even cooler, and see what you can find in your own log files.

|  |  |
| --- | --- |
| Log | Search Term |
| Any | NodeRoleWatcher  Across all the processes, watch as they detect changes in the processes that are running in the cluster, and see what they do about it! |
| Coordinator / Overlord | org.apache.druid.metadata.SQLMetadataRuleManager  This is the coordinator polling the rules in the metadata database, getting ready to apply them. The log tells you how many rules it picks up and how many data sources they cover. |
| Coordinator / Overlord | org.apache.druid.metadata.SqlSegmentsMetadataManager  Messages show how many segments the cluster thinks are “used” – ready to be used for queries. |
| Coordinator / Overlord | org.apache.druid.indexing.overlord.RemoteTaskRunner  GIves interesting information about what’s happening with ingestion resources in the cluster, including when they first advertise themselves. |
| Coordinator / Overlord | org.apache.druid.server.coordinator.rules  Lots of information about how the retention rules are actually being applied (or not!). |
| Historical | org.apache.druid.server.coordination.BatchDataSegmentAnnouncer  You can see individual segments being announced as available for query by each historical server as it loads them. |
| HIstorical | org.apache.druid.server.coordination.SegmentLoadDropHandler  As well as seeing how the historical checks its local segment cache on startup, you can watch along as the Historical picks up the instructions from the coordinator and then does something about them. When there are ERRORs like “Failed to load segment for dataSource” you get traces about what the process was trying to do – quite often something pointing to an error with its connection to deep storage. |
| Any | org.apache.druid.initialization.Initialization  These messages are all about the process starting up. It can be interesting to see what exactly each one does – and if it runs into issues. |
| Coordinator | org.apache.druid.server.coordinator.duty.BalanceSegments  Here you can see what Druid decides to do when balancing needs to be carried out – e.g. a server is lost or added. |