Logging

Introduction

This page provides a narrative on Nutch logging. Nutch uses Simple Logging Facade for Java (SLF4J) API's and Apache Log4j 2 as the logging implementation.

Audience

The page is targeted towards

- users who wish to learn about default logging in Nutch
- developers who would wish to further extend/customize logging

Related Development Work

- **NUTCH-2885** - Getting issue details... STATUS (pull request also linked)

Example Logging Syntax

src/java/org/apache/nutch/crawl/Injector.java

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
...
public class Injector extends NutchTool implements Tool {
  private static final Logger LOG = LoggerFactory
      .getLogger(MethodHandles.lookup().lookupClass());
...
    @Override
    public void setup(Context context) {
      ...
      LOG.info("Injector: overwrite: " + overwrite);
      LOG.info("Injector: update: " + update);
    }
```

Default Configuration

> **Legacy logging**
>
> Prior to Nutch version 1.19, Nutch logging was configured via *conf/log4j.properties* this changed in Nutch 1.19... see below

As of version 1.19 Nutch uses **conf/log4j2.xml** for logging configuration. By default, Nutch will log to log to ***$NUTCH_HOME/logs/hadoop.log***

The configuration uses a RollingFileAppender with the cron triggering policy configured to trigger every day at midnight. Archives are stored in a directory based on the current year and month. All files under the base directory that match the `*/nutch-*.log.gz` glob and are 60 days old or older are deleted at rollover time. Additionally, it uses the ConsoleAppender configuration so everything is also written to STDOUT. This is useful for things like the ParserChecker and similar tooling.

Extending Nutch Logging Configuration

Log4j2 provides many Appenders which can be configured to extend Nutch logging. See below for some examples of how this could be done

logzio-log4j2-appender

The Logzio Log4j 2 Appender ships logs to Logzio using HTTPS bulk. It can be configured as follows

Add a dependency to ivy/ivy.xml

ivy.xml

```
<dependency org="io.logz.log4j2" name="logzio-log4j2-appender" rev="1.0.13" conf="*->master" />
```

Augment the log4j2.xml configuration

log4j2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
...
<Configuration status="info" name="Nutch" packages="">
...
  <Appenders>
    <LogzioAppender name="Logzio">
        <addHostname>true</addHostname>
        <logzioToken>${insert_your_token_here}</logzioToken>
        <logzioType>java</logzioType>
        <logzioUrl>https://listener.logz.io:8071</logzioUrl>
    </LogzioAppender>
...
  </Appenders>
  <Loggers>
    <Root level="info">
      <AppenderRef ref="Logzio"/>
...
    </Root>
  </Loggers>
</Configuration>
```

Logging to Splunk Enterprise Server

This demonstrates how to log events to the Splunk HTTP Event Collector or to a TCP input on a Splunk Enterprise instance.

Augment log4j2.xml with the following

```
<?xml version="1.0" encoding="UTF-8"?>
...
<Configuration status="info" name="Nutch" packages="">
...
    <!-- Define an appender that writes to a TCP socket. We use Log4J's SocketAppender, which
         is documented at
             https://logging.apache.org/log4j/2.x/manual/appenders.html#SocketAppender
         Note that TCP inputs are *not* the same as Splunk's management port.
    -->
    <Appenders>
        <Socket name="socket" host="${insert_splunk_host}" port="${insert_splunk_port}">
            <PatternLayout pattern="%p: %m%n" charset="UTF-8"/>
        </Socket>

        <SplunkHttp name="http-input"
              url="${insert_splunk_host}:${insert_splunk_port}"
              token="${insert_splunk_token}"
              host=""
              index=""
              source="..."
              sourcetype="..."
              messageFormat="text"
              middleware="HttpEventCollectorUnitTestMiddleware"
              batch_size_bytes="0"
              batch_size_count="0"
              batch_interval="0"
              connect_timeout="5000"
              disableCertificateValidation="true">
            <PatternLayout pattern="%m"/>
        </SplunkHttp>
    </Appenders>
    <Loggers>
        <Root level="INFO">
        </Root>
        <Logger name="splunk.logger" level="info">
            <AppenderRef ref="socket"/>
        </Logger>
        ...
        <Logger name="splunk.log4j" level="info">
            <AppenderRef ref="http-input"/>
        </Logger>

    </Loggers>

</Configuration>
```