

Poi Utils

Overview

by Nicola Ken Barozzi, Andrew C. Oliver

1. Logging

Logging in POI is used only as a debugging mechanism, not a normal runtime logging system. Logging is **ONLY** for autopsy type debugging, and should **NEVER** be enabled on a production system. Enabling logging will reduce performance by at least a factor of 100. If you are not developing POI or trying to debug why POI isn't reading a file correctly, then **DO NOT** enable logging. You've been warned.

Hence, we need to be able to easily disable it entirely and make POI not dependent on any logging package.

Note:

POI is not dependent on commons-logging for running, but not for compiling.

1.1. Logging Overview

Every class uses a `POILogger` to log, and gets it using a static method of the `POILogFactory`.

The `POILogFactory` uses the `NullLogger` by default; it can be instructed to use any other `POILogger` implementation by setting the system property `org.apache.poi.util.POILogger`.

Note:

```
java -Dorg.apache.poi.util.POILogger=the.package.of.MyPoiLoggerImpl ProgramThatUsesPoi
```

FIXME (nicolaken):

Still needs testing.

1.2. POILogFactory

Each class in POI can get its POILogger by calling a static method of the POILogFactory.

1.3. POILogger

Each class in POI can log using a POILogger, which is an abstract class. We decided to make our own logging facade because:

1. we need to log many values and we put many methods in this class to facilitate the programmer, without having him write string concatenations;
2. we need to be able to use POI without any logger package present.

There are three implementations available, and you can roll out your own, just extend `org.apache.poi.util.POILogger`.

1.3.1. NullLogger

Discards every logging request.

1.3.2. SystemOutLogger

Sends every logging request to `System.out`.

1.3.3. CommonsLogger

Sends every logging request to the Commons Logging package. This can use JDK1.4 logging, log4j, logkit, and is an actively maintained Jakarta Project.