A sample C# project demonstrates how to perform semantic logging in any .NET application.

Semantic logging (aka Structured logging) goes beyond outputting text strings and allows rich data to be captured to the logging back-end. When used in conjunction with a semantic logging service such as Seq, the usefulness of log output is dramatically increased. It can also be used to capture diagnostic data from production code, which can be viewed in real time on the server or queried later using a SQL-like syntax. There is a reasonably good video examining some of the pros and cons of logging techniques at <https://youtu.be/NlBjVJPkT6M?si=bYWm7O13ptlfGs_l&t=1409> (Nick Chapsas, NDC Oslo 2023).

The NuGet package TA.Utilities (open source, MIT license) defines an abstract Ilog interface along with IFluentLogBuilder that is very similar to the one used by NLog. TA.Utilities.Logging.NLog contains an implementation that uses NLog as the rendering service. The point of using an abstract interface is that it avoids taking a hard dependency on the logging back-end, so it would be relatively easy to produce an adapter for Serilog, Log4Net, etc. The actual logging service can be “plugged in” at runtime.

An additional NuGet package, NLog.Targets.Seq, provides a logging target that ships log records to a remote server. This is done asynchronously and has no noticeable effect on application performance, even if the server goes offline. Seq is very powerful and allows real-time viewing, querying of historical data using SQL queries, creation of dashboards and alerts and other good stuff. It is free for a single user and can be easily set up in a Docker container.

Note that it is possible to render logs to multiple targets, so it is perfectly feasible to create a log file on disk while also sending logs to a Seq server and displaying them on the console, all at potentially different verbosity levels. It’s very flexible and is configured through a text file that can even be changed while the program is running.

The sample C# project is available on BitBucket at <https://bitbucket.org/oceansoftware/osl.patternsandpractices.livelogging>

About Seq

Seq is a real-time search and analysis server for structured application logs. Its carefully designed user interface, JSON event store, and familiar query language make it an efficient platform for detecting and diagnosing issues in complex applications and microservices. Seq is self-hosted and runs on Windows or under Docker/Linux.

The setup is relatively straightforward, requiring the installation of one NuGet package (and its dependencies) and a NLog.config configuration file.

In the root directory of the repository there is a ReadMe.md file with more detailed information.

## Logging Best Practices

* Unhandled Exceptions: Install a last-chance exception handler in your .NET application so that you can log the exception before the application crashes. This can be extremely useful when debugging unexplained crashes that don’t appear to generate any error messages. Some code like this in your Program.cs class