Implementing Logging

Production Insights Using Logging

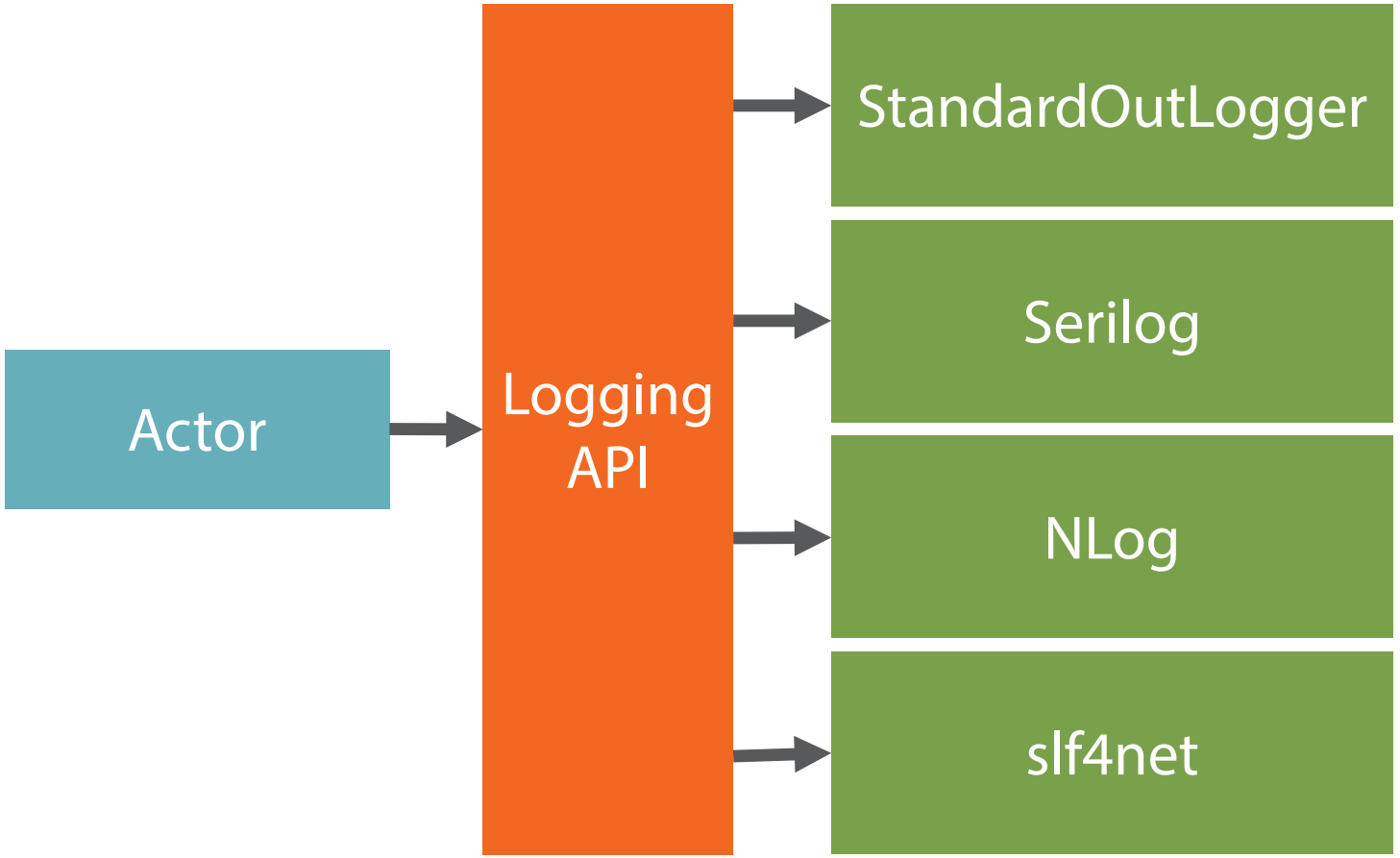
Jason Roberts

@robertsjason | dontcodetired.com

Overview

- Log output choices and logging levels
- HOCON configuration for logging
- Built-in StandardOutLogger
- Using NLog to write to log files
- Using Serilog to write to Seq

Logging Choices



Color coded console output

Files, SQL Server, RavenDB, Raygun, Windows Event Log, Seq Server, Elasticsearch, etc.

Files, SQL Server, Windows Event Log, Email, MSMQ, ASP.NET Trace, etc.

Wrappers for NLog, log4net, custom wrappers

Logging Levels

Because Akka.NET systems are designed to be self healing, consider what constitutes an “error” log event

Logging Levels

- Errors
  - Conditions within the actor system requiring human interaction to fix
  - Errors in dependent resources (e.g. database) that need human interaction to fix
  - Serious problems triggering out-of-hours callout
  - Developers / operations
- Warnings
  - Expected errors that have been handled by supervisors
  - Conditions that may lead to errors in the future
  - Doesn't trigger out-of-hours callout (e.g. checked daily by operations / developers)
  - Developers / operations

Logging Levels

- Info
  - Useful information / insights into the running of the actor system
  - E.g. user starting to watch a movie
  - Doesn't trigger out-of-hours hours callout
  - Developers / operations / business
- Debug
  - Low level details (e.g. actor lifecycle events)
  - Fault diagnosis, fault resolution, development time insights
  - Doesn't trigger out-of-hours hours callout
  - Developers / operations

HOCON Configuration

```
<akka>
   <hocon>
     <![CDATA[
          akka {
               loglevel = INFO
          }
     ]]>
   </hocon>
 </akka>
```

Getting Started with the StandardOutLogger

Add ILoggingAdapter field _logger

_logger.Info(…)

_logger.Warning(…)

Writing Error Log Messages

Add logging to PlaybackStatisticsActor

_logger.Error(exception, …)

Writing to Log Files with NLog

(all TODOs done)

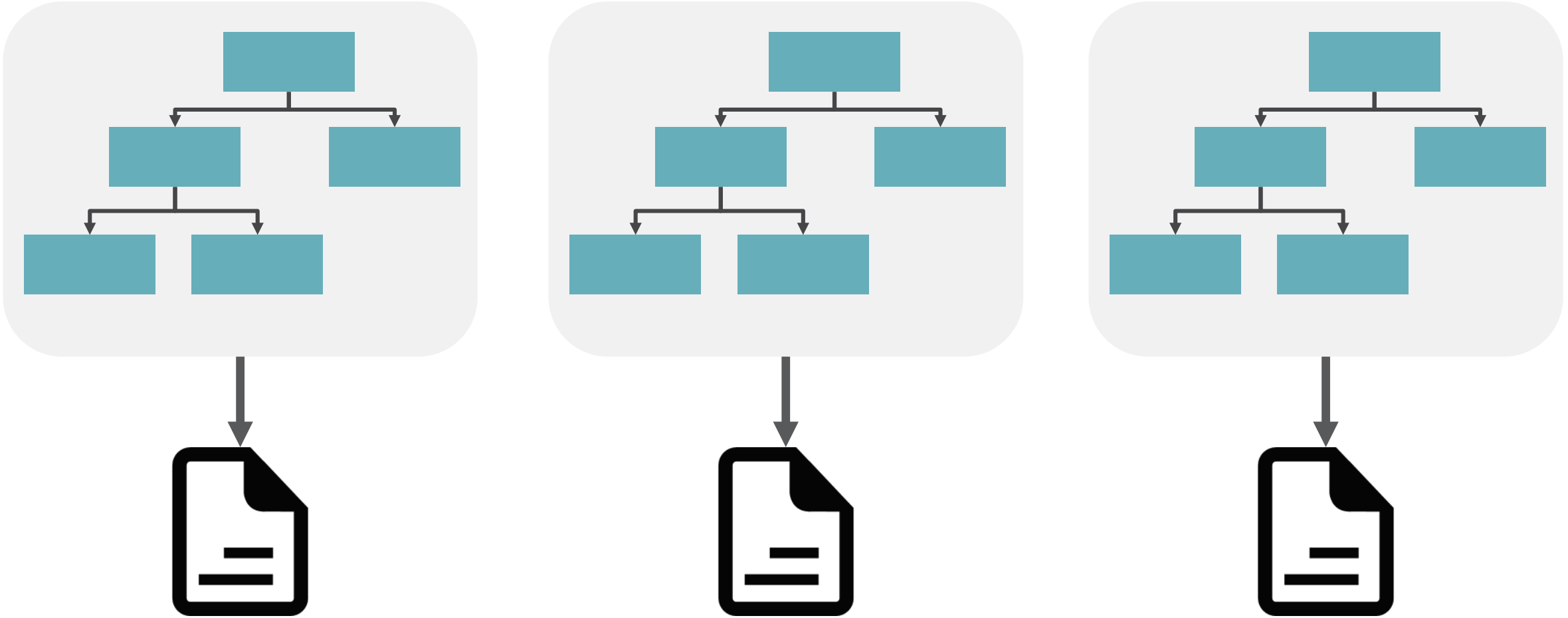
Install Akka.Logger.Nlog package

Modify HOCON config

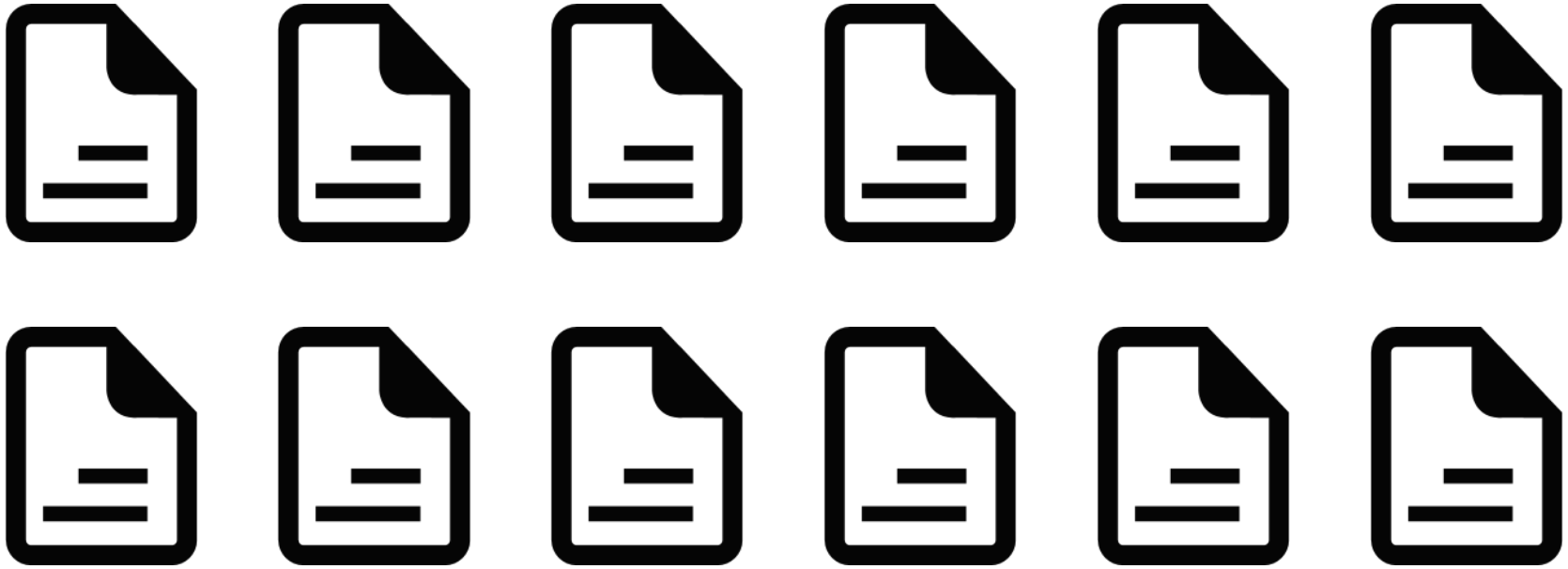
Add NLog.config & set copy to output

Configure NLog to write to a file

Distributed Logging



Distributed Logging



Generic log data stores

- SQL Server
- RavenDB
- MongoDB
- Etc.

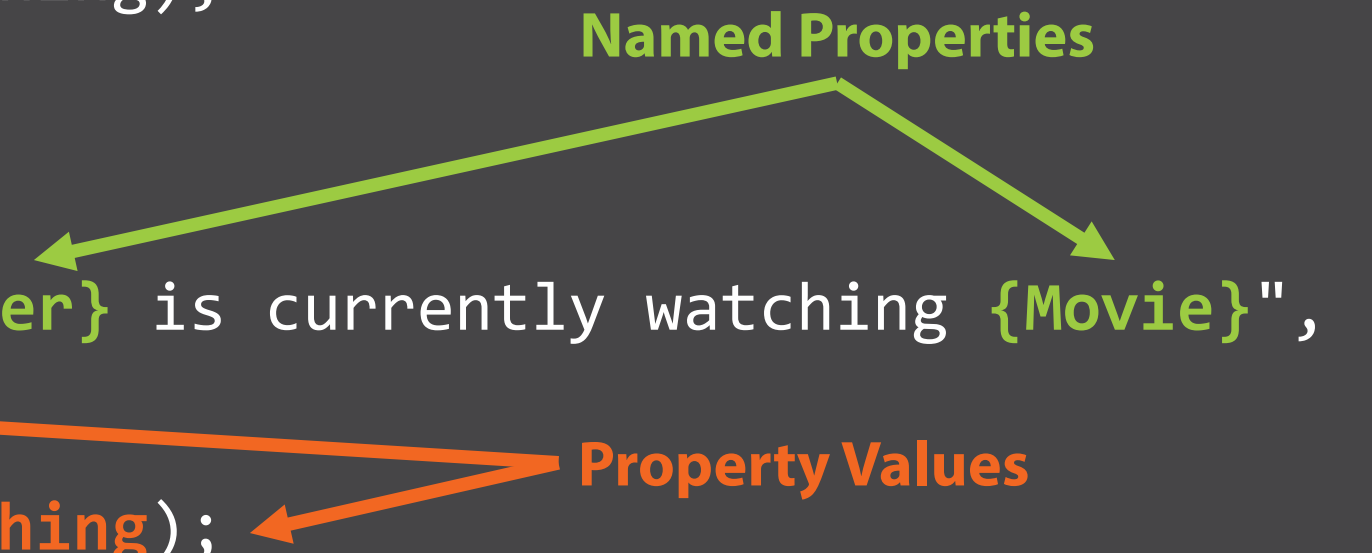
"Tailored" log data stores

- Elastic search
- Seq

Serilog Overview

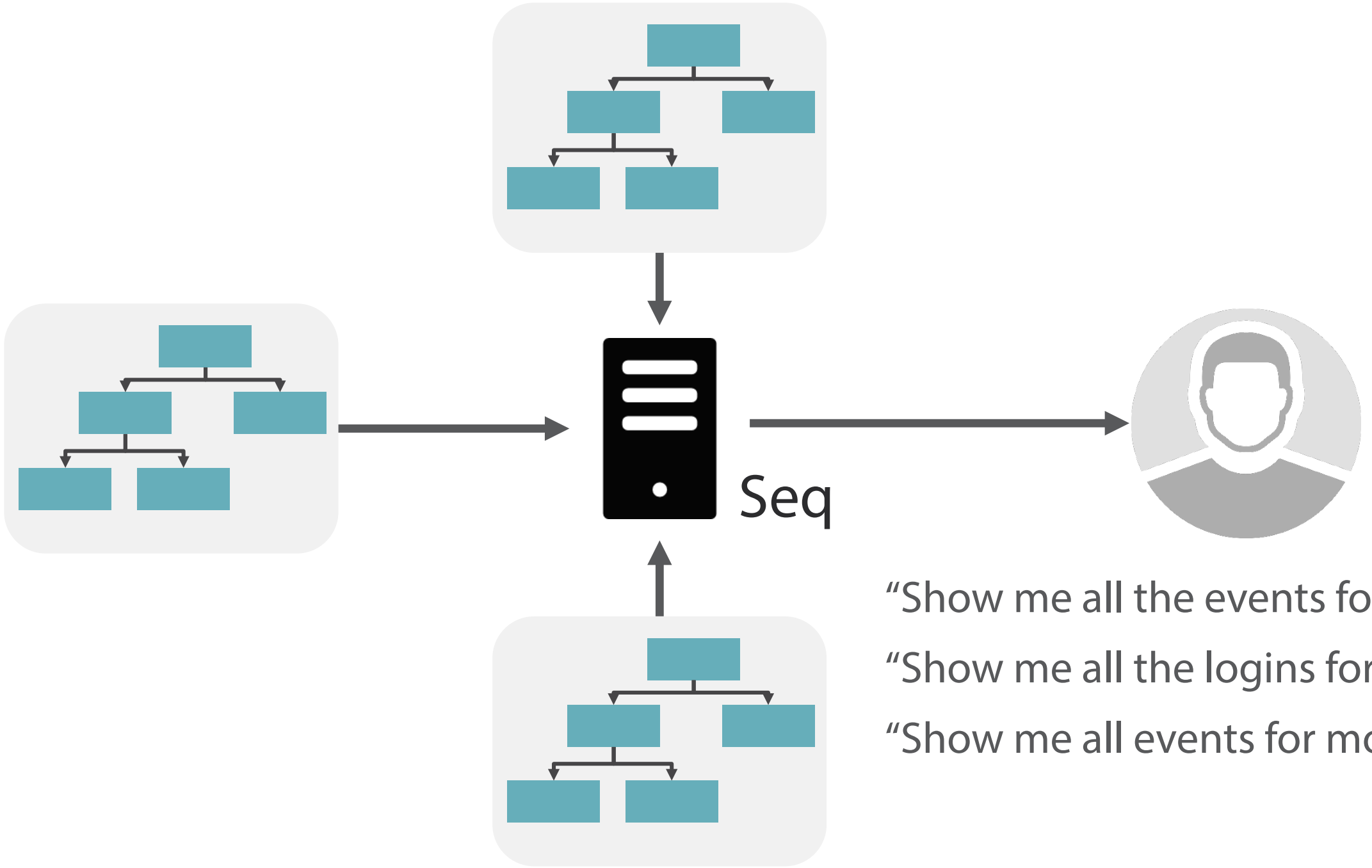
```
_logger.Info("UserActor {0} is currently watching {1}",
             _userId,
             _currentlyWatching);
```

```
_logger.Info("UserActor {User} is currently watching {Movie}",
             _userId,
             _currentlyWatching);
```



Pluralsight course:

"Modern Structured Logging
With Serilog and Seq"



"Show me all the events for user 42"

"Show me all the logins for all users"

"Show me all events for movie 'x'"

Implementing Serilog Logging

(all actors changed)

Install Akka.Logger.Serilog package

Modify HOCON config

Add Serilog configuration code

Configure Serilog to write to Seq

Query Seq

Automatic Debug Logging

```
akka {
    loglevel = DEBUG

    loggers = ["Akka.Logger.NLog.NLogLogger, Akka.Logger.NLog"]

    actor {
        debug {
            receive = on          # log any received message
            autoreceive = on      # log automatically received messages, e.g. PoisonPill
            lifecycle = on        # log actor lifecycle changes
            event-stream = on     # log subscription changes for Akka.NET event stream
            unhandled = on        # log unhandled messages sent to actors
        }
    }
}
```

Summary

Log output choices

Debug, Info, Warning, Error levels

Built-in StandardOutLogger

loggers = ["Akka.Logger.Serilog.SerilogLogger, Akka.Logger.Serilog"]

Using NLog to write to log files

Using Serilog to write to Seq

Automatic debug logging configuration

Next:

Implementing Dependency Injection