Design Document



**Application Logger**

Version: 1.0

Mar 2016

Table of Contents

[1. Introduction 3](#_Toc447034794)

[1.1 Purpose 3](#_Toc447034795)

[1.2 Scope 3](#_Toc447034796)

[1.3 Document Organization 3](#_Toc447034797)

[1.4 Audience 4](#_Toc447034798)

[1.5 Acronyms, Abbreviations, Terms and Definitions 4](#_Toc447034799)

[2. Design Overview 5](#_Toc447034800)

[2.1 Approach 5](#_Toc447034801)

[2.2 Architectural Goals and Constraints 5](#_Toc447034802)

[2.3 Design Principles 5](#_Toc447034803)

[3 Topology Diagram 6](#_Toc447034804)

[4 Application Architecture 6](#_Toc447034805)

[4.1 Common Applications Framework 7](#_Toc447034806)

[4.2 Rules Engine Design 7](#_Toc447034807)

[4.3 Sequence of Events 7](#_Toc447034808)

[4.4 Object Model 7](#_Toc447034809)

[5 Application Implementation 7](#_Toc447034810)

[6 Database Architecture 7](#_Toc447034811)

[6.1 Data Model 7](#_Toc447034812)

[6.2 Tables 7](#_Toc447034813)

[7 Assumptions and Constraints 7](#_Toc447034814)

[Appendix A: Acronyms, Abbreviations, Terms and Definitions 8](#_Toc447034815)

[Appendix B: Products & Tools 8](#_Toc447034816)

[Appendix C: Configuration files 8](#_Toc447034817)

[Appendix C: Configuration files 8](#_Toc447034818)

# Introduction

## 1.1 Purpose

The purpose of this document is to outline the technical design of the Application Logger tool. Its main purpose is to –

* Provide the link between the Functional Specification and the detailed Technical Design documents
* Detail the functionality which will be provided by each component or group of components and show how the various components interact in the design
* Provide a basis for the Application Logger tool’s detailed design and development

This document is not intended to address installation and configuration details of the actual implementation. As is true with any high level design, this document will be updated and refined based on changing requirements.

## 1.2 Scope

The Application Design outlined in this document builds upon the scope defined in the Requirements phase.

## 1.3 Document Organization

This document is organized into the following sections:

|  |  |
| --- | --- |
| Introduction | Provides information related to this document (e.g. purpose, term definitions etc.) |
| Design Overview | Describes the approach, architectural goals and constraints, Guiding principles, Java Design patterns used in design and development |
| Topology Diagram | Describes the various system components and the integration between them |
| Application Architecture | Describe the application architecture in terms of different layers of application. Description of the presentation layer, business layer, data access layer and resource layer and their relationship to each other. |
| Object Model | Describes the conceptual representation of the problem domain of an application that embodies the business rules being automated and is usually represented with Class diagram |
| Database Architecture | Describes the overall Data model for the screening tool |
| Assumptions and Constraints | Details various assumptions made during design and development of the Online Screening too |
| Appendix A | Describes Acronyms, Abbreviations, Terms and Definitions |
| Appendix B | Lists all products and tools used in design and development |
| Appendix C | Lists all the configuration files used in implementation |
| Appendix D | Describes the data dictionary |

## 1.4 Audience

The intended audiences for this document are the project development teams, technical architects, database designers, testers and vendors.

## 1.5 Acronyms, Abbreviations, Terms and Definitions

Please refer to Appendix A for a list of all acronyms and abbreviations.

# Design Overview

# 

## 2.1 Approach

Approach This document is created and extended in multiple phases over the course of the project

* Requirements Phase - During the Requirements Phase, the initial version of this document is created, describing the candidate architecture to be validated in the System Design Phase.
* System Design Phase - During the System Design phase, the Evolutionary Prototype is created and this document is finalized by establishing a sound architectural foundation for the Construction Phase.
* Construction Phase – During the Construction Phase, this document is not expected to change radically; it is mainly updated to reflect changes in any interface definitions.
* Transition / Training Phase – During the Transition/Training Phase, no further additions or modifications are made to this document.

## 2.2 Architectural Goals and Constraints

The overall architecture goal of the system is to provide a highly available and scalable tool that provide functionality to logs message of application to console, file and database. By help of this tool we can log error, debug and information messages.

Destination

Level Object

Info is

Info is logged

Log Manager

## Logged

Logging Information

Logging Information

Appender Object

Logger Object

Formatted

Info

Filter Object

Formatted

Info

Layout Object

Object Renderer

Core Objects

Java Program

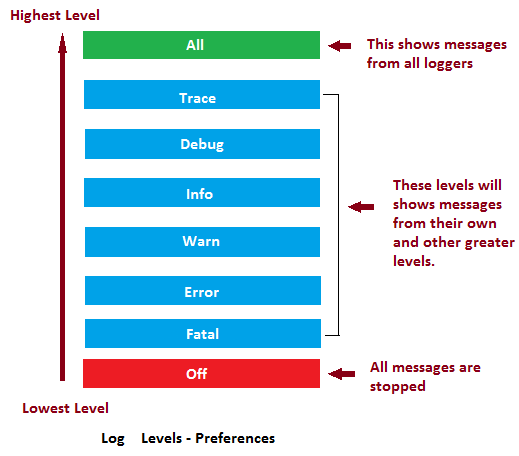
Support object

Fig.1.0 Architecture of Logger Application

## 2.3 Design Principles

Best practices and design principles will be applied in two main areas –

* Basic principal of Application Logger is to inform that, how is our application running on server and information about which part of our application is currently running.
* By Application Logger we also get the information about level (type) of message like error, debug, fetal, info, warn, trace.



* Logger has to log message to console, File, Database. These functionality of logger give an opportunity to put messages into console, file and database.

Figure 2.0 Application logger class

* Logger has filter function to control all these messages. We can choose the maximum and minimum level of log message.

Input = All messages

# 3 Topology Diagram

The diagram below provides an illustration of the System Architecture along with various system components that will be used in architecting the Application Logger Tool-

Appender Object

Logger Object

Database

Appender

File Appender

Console

Appender

Log message

Database

File

Console

Filter

Layout

Fig. Topology Diagram

Log message – Application information message.

Logger Object – Main object that send message to Appender.

Filter – Filter the message from all messages present into logger.

Layout - It set format of output message

Console Appender - Append message to console

File Appender - Append message to File

Database Appender - Append message to Database

# 4 Application Architecture

Application architecture defines the various components and their interactions in context of a whole system. Application architecture is the critical software that bridges the architectural gap between the application server and the application’s business logic, thereby eliminating the complexities and excessive costs of constructing, deploying and managing distributed enterprise applications.

## 4.1 Common Applications Framework

## 4.2 Rules Engine Design

## 4.3 Sequence of Events

1. We have to create Logger Object.
2. Then we set parameter of logger like

* Name
* Layout
* File Name
* Target Stream

1. Set Appender means we have to add Appender (Console, File, Database) into logger object.
2. Now Done. We can use this logger object to log Application information.

# 5 Application Implementation

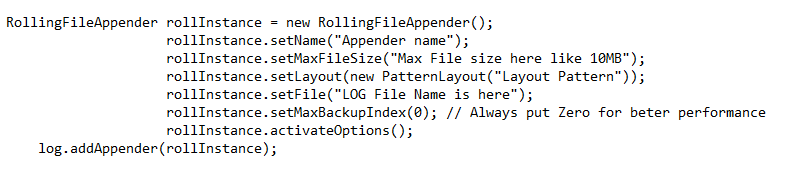
If you want to use this logger tool in your application then you need to follow these steps

Step 1. First import Log4j.jar and Application Logger Tool.jar.

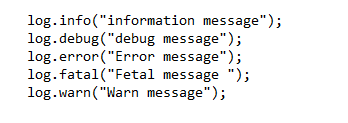
Step2. Create a logger object



Step3. Create Appender Object then add into logger object.

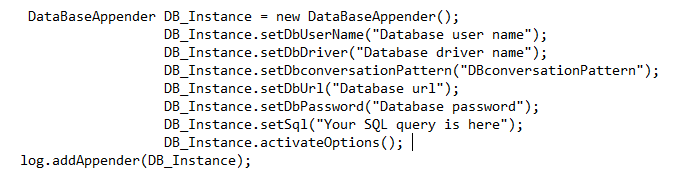


Step 4. Your Logger is ready. Use it.



# 6 Database Architecture

Database is Optional for this application. Application Logger tool needed database only when we want to put log message into database. If want then create a DatabaseAppender object set parameter and add it to logger object.



It is good if you add filter in Database Appender to reduce load of database. By using Filter object we can set type of message which we want to store in database. To add filter follow some steps,

1. Create object of LevelRangeFilter class



1. Set maximum and minimum level of message by using Level Literals.



1. Add this filter object to Appender object.

To add filter in File Appender



To add filter in Database Appender



## 6.1 Data Model

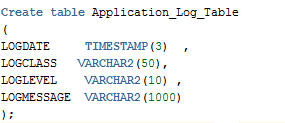
A data model is an abstract model that organizes elements of data and standardizes how they relate to one another and to properties of the real world.

## 6.2 Tables

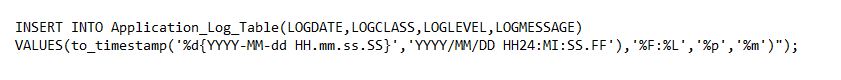
Tables are organized into rows and columns.

## Table Configuration

Before you start using JDBC based logging, you should create a table to maintain all the log information. Following is the SQL Statement for creating the LOG table –



In application log table LOGDATE column should be TIMESTAMP and SQL query should be like this-



# 7 Assumptions and Constraints

* We assume that program runs in normal configuration.
* Operating System – windows 7,8,10 Linux ,OSX
* Minimum 2 GB of Ram
* Java Runtime Environment – 8.0 and above.
* Database supported – MySQL, Oracle 11g.

# Appendix A: Acronyms, Abbreviations, Terms and Definitions

### Logger Object

The top-level layer is the Logger which provides the Logger object. The Logger object is responsible for capturing logging information and they are stored in a namespace hierarchy.

### Layout Object

The layout layer provides objects which are used to format logging information in different styles. It provides support to Appender objects before publishing logging information.

Layout objects play an important role in publishing logging information in a way that is human-readable and reusable.

### Appender Object

This is a lower-level layer which provides Appender objects. The Appender object is responsible for publishing logging information to various preferred destinations such as a database, file, console, UNIX Syslog, etc.

### Level Object

The Level object defines the granularity and priority of any logging information. There are seven levels of logging defined within the API: OFF, DEBUG, INFO, ERROR, WARN, FATAL, and ALL.

### Filter Object

The Filter object is used to analyze logging information and make further decisions on whether that information should be logged or not.

An Appender objects can have several Filter objects associated with them. If logging information is passed to a particular Appender object, all the Filter objects associated with that Appender need to approve the logging information before it can be published to the attached destination.

# Appendix B: Products & Tools

# Appendix C: Configuration files

Application does not required any configuration file but you use Database Appender then program required database configuration file to configure database parameter.