**What is Logging?**

Logging is a process that takes applications to a newer level with information logs on how the applications may or may not have performed/executed. It gives an exact idea of software performance, including any shortcomings.

Log4j in [Selenium](https://www.browserstack.com/selenium) is one such logging framework that helps in gathering information in the form of logs or log files.

**What is Log4j in Selenium?**

Log4j is a logging framework written in Java that provides an easy way for logging in Selenium. In a nutshell, the framework gives out information about everything that goes on during the software execution.

Log4j also provides insight into anything that may have gone wrong during software execution or automation. Overall, Log4j documents the output in the form of logs that can be examined later for purposes such as auditing small and large-scale [Selenium projects](https://www.browserstack.com/guide/build-and-execute-selenium-projects).

Let’s take a look at the various components of the Log4j logging framework.

**Components of Log4j**

The Log4j logging framework comprises the following components:

1. Logger
2. Appenders
3. Layout

**Logger**

The function of the logger in Log4j is basically storing and capturing all the necessary logging information that will be generated using the framework.

To truly understand its functioning, let’s dig a little deeper and discuss the logger class, and log level methods. The loggers also decide which priority is going to be captured.

1. **Logger Class** – To fully use the logger, create an instance for a logger class where all the generic methods will be at the user’s disposal, required to use Log4j.
2. **Log Levels** – These are the methods that will be used to print the log messages. There are primarily only a few log levels that are used in a script.

* **ALL –** This level will prioritize and include everything in the logs.
* **ERROR –** This level will show messages that inform users about error events that may not stop the application.
* **WARN –** This level will show information regarding warnings, that may not stop the execution but may still cause problems.
* **DEBUG –** This level will log debugging information.
* **INFO –** This level will log the progress of the application.
* **FATAL –** This will print information critical to the system that may even crash the application.

**Appenders**

The appender basically grabs information from the logger and writes log messages to a file or any other storage location. The following are some of the appenders one can use for Log4j:

1. **FileAppender –** This will append the log messages to a file.
2. **RollingFileAppender –** It will perform the same function as FileAppender, but users will be able to specify the maximum file size. Once the limit is exceeded, the appender will create another file to write the messages.
3. **DailyRollingFileAppender –** It specifies the frequency by which the messages are to be written to the file.
4. **ConsoleAppender –** In this, the appender will simply write the log messages in the console.

**Layout**

The layout is where the format in which log messages will appear is decided. There are several layouts one can use for log messages:

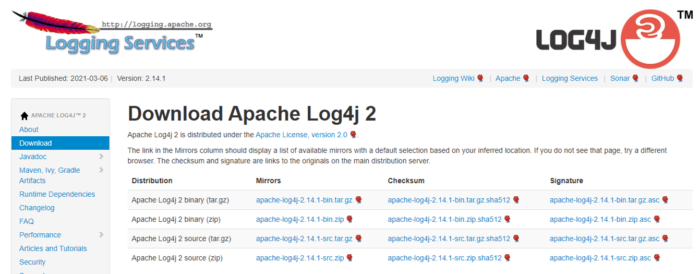
1. **Pattern Layout –** The user must specify a conversion pattern based on which the logs will be displayed. Otherwise, it takes the default conversion pattern in case of “no pattern specified”.
2. **HTML Layout –** In this layout, the format of logs will be in the form of an HTML table.
3. **XML Layout –** This will show the logs in an XML format.

Now, let’s take a look at how to download and set up Log4j.

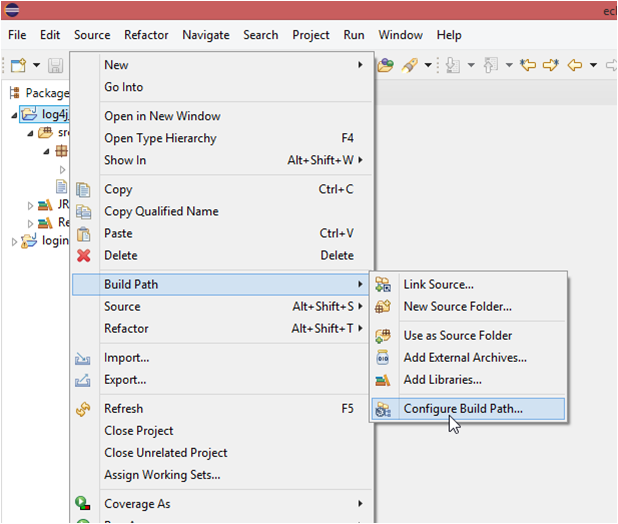
**How to Set up Log4j in Selenium**

To efficiently download and set up Log4j for Selenium, follow the approach below:

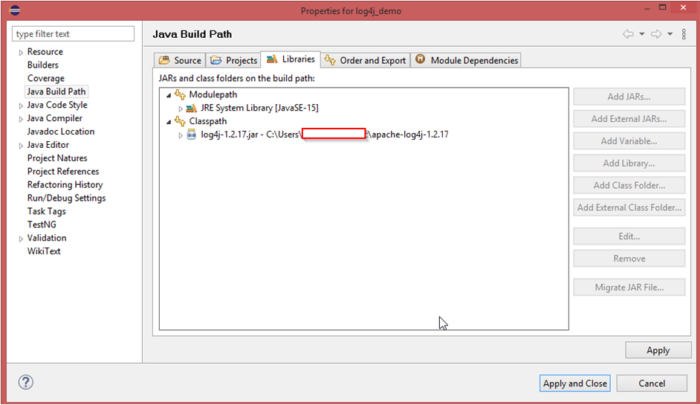
1. Download Log4j from the official [Apache website](https://logging.apache.org/log4j/2.x/).



2. After downloading the **Log4j,** add the **log4j.jar** file to the java build path.



3. After adding the jar file in the configuration, one can use **Log4j** successfully in scripts.



Now, let’s understand how to use this logging framework in a program.

**How to Use Log4j in Selenium**

Follow the steps below to successfully run Log4j with [Selenium Webdriver](https://www.browserstack.com/guide/selenium-webdriver-tutorial):

1. Write an automation script, such as the one in the example below. It is a simple script that opens a URL, send keys to username and password, and ends the script when it clicks the login button.
2. After creating the script, create a **log4j.properties** file and specify the root logger, appended, and layout information in the file.
3. Import log4j dependencies like **Logger, PropertyConfigurator,** and add them to the script along with the logger class.
4. Add the messages that will be displayed in the log file.

Now, let’s try to understand the code written below:

[java]

package newpackage;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeOptions;

import org.openqa.selenium.UnexpectedAlertBehaviour;

import org.openqa.selenium.chrome.ChromeDriver;

import org.apache.log4j.Logger;

import org.apache.log4j.PropertyConfigurator;

import org.openqa.selenium.JavascriptExecutor;

public class Myclass {

static Logger log = Logger.getLogger(Myclass.class);

public static void main(String[] args) {

PropertyConfigurator.configure("path\\to\\log4j.properties");

System.setProperty("webdriver.chrome.driver", "Path\\to\\chromedriver.exe");

ChromeOptions options = new ChromeOptions();

options.setUnhandledPromptBehaviour(UnexpectedAlertBehaviour.IGNORE);

WebDriver driver = new ChromeDriver(options);

JavascriptExecutor js = (JavascriptExecutor)driver;

driver.get("https://www.browserstack.com/users/sign\_in");

log.info("Open browserstack");

driver.manage().window().maximize();

log.info("Maximize window size");

js.executeScript("document.getElementById('user\_email\_login').value='rbc@xyz.com';");

log.info("enter username");

js.executeScript("document.getElementById('user\_password').value='password';");

log.info("enter password");

js.executeScript("document.getElementById('user\_submit').click();");

log.info("click on login");

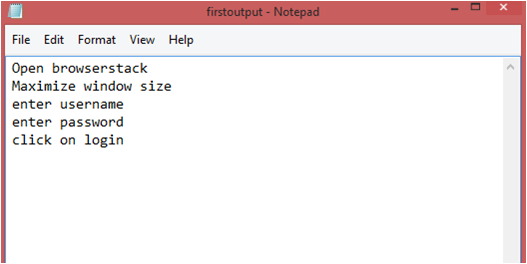
driver.close();

}

}

[/java]

**Output:**



The above program will run an automation script, and simultaneously create a log file at the specified location with the name given in the **log4j.properties** file.

Now, let’s understand how to create the **log4j.properties** file.

**Log4j with Properties File Configuration**

The properties configuration file will contain the information regarding the following:

1. The first objective is to identify the log level and the destination where the log will be written. In this case, the log level is the **DEBUG level** and a file is the destination.
2. The next step is to establish an appender. This example uses a **RollingFIleAppender** with a maximum capacity of 2MB and the file location is also specified in this section. The **MaxBackupIndex** is the number of files that will be created in case the file size is exceeded.
3. The last section contains the layout in which a format is specified. This example uses **PatternLayout**. And, since we have not used a conversion pattern here, it will take the default conversion pattern into account.

[code]

#root logger

log4j.rootLogger = DEBUG, file

#appender

log4j.appender.file = org.apache.log4j.RollingFileAppender

log4j.appender.file.File = PATH\\TO\\firstoutput.log

log4j.appender.file.MaxFileSize = 2MB

log4j.appender.file.MaxBackupIndex = 3

#layout

log4j.appender.file.layout = org.apache.log4j.PatternLayout

[/code]

**Why Use Log4j in Selenium?**

Use Log4j logging framework in Selenium for the following reasons:

1. Log4j logging framework can help in debugging applications easily. With different log levels, it becomes easier to sort information based on categories.
2. The logging framework is open source and can help in logging in different log levels and suppress logs in different environments – which ultimately improves application performance.
3. The framework produces better output in general.
4. With three components and clarity in usage, it becomes easier to use and configure log4j in Selenium.
5. The setup is easy and free of cost, making it more accessible for faster debugging.