http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte3:fdl:logging:log4j_2:%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D%EB%82%B4%EC%97%90%EC%84%9C_%EC%A7%81%EC%A0%91_%EC%84%A4%EC%A0%95%ED%95%98%EB%8A%94_%EB%B0%A9%EB%B2%95

Log4j 2 Configurations [configuration by coding]

Outline

This Section describes how to configure Log4J 2 properties (Appender, Layout, Log Level, etc.) by coding.
You need no configuration files when you work on configuration by coding.

Description

Applications

You can obtain the logger objects in the code without Log4J 2 Configurations.
Create the logger objects using the Method LogManager.getLogger(). Note that Log4j 2 returns the default logger object.
Refer to the following example for how the default logger object has been made:

```
Log Level : ERROR
Appender    : ConsoleAppender
Layout       : PatternLayout
pattern      : %d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n
```

You can configure the Log Level, Appender, Layout and other properties by casting the foregoing logger object to org.apache.logging.log4j.core.Logger.
See the following for the configuration method used for the foregoing example: You can check out Log4j 2 API for more information:

```
LogManager.getLogger()                                                    -- Create Logger
Logger.addAppender(), Logger.removeAppender(), Logger.getAppenders()    -- Add, remove and acquire Appenders to Logger.
Layout.createLayout()                                                     -- Create Layout
Appender.createAppender(), Appender.removeAppender()                      -- Generate and delete Appender
Logger.setLevel(), Logger.getLevel()                                      -- Configure and obtain Log Level
```

Application

@Test

```
public void log4j2ConfigTest() {

		// Generate the default Logger
		Logger logger = (Logger) LogManager.getLogger();

		// Verify the default configuration
		// Log Level: ERROR
		assertEquals(Level.ERROR, logger.getLevel());
		// Appender: Console
		Map<String, Appender> appenders = logger.getAppenders();
		assertEquals(1, appenders.size());
		assertEquals(ConsoleAppender.class, appenders.get("Console").getClass());
		// Layout: Pattern
		ConsoleAppender console = (ConsoleAppender) appenders.get("Console");
		assertEquals(PatternLayout.class, console.getLayout().getClass());
		PatternLayout pattern = (PatternLayout) console.getLayout();
		assertEquals("%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n",
pattern.toString());

		// Original: Logs above ERROR Level
		logger.debug("Before: [DEBUG] Test log4j 2.");
		logger.info("Before: [INFO] Test log4j 2.");
		logger.warn("Before: [WARN] Test log4j 2.");
		logger.error("Before: [ERROR] Test log4j 2.");
		logger.fatal("Before: [FATAL] Test log4j 2.");

		/*
		Logs
		===================
		14:17:09.930 [main] ERROR egovframework.rte.fdl.logging.LogTest - Before:
[ERROR] Test log4j 2.
		14:17:09.931 [main] FATAL egovframework.rte.fdl.logging.LogTest - Before:
[FATAL] Test log4j 2
		*/

		// Changing default configuration
		logger.removeAppender(console);
		// Appender: File
		PatternLayout layout = PatternLayout.createLayout("%m%n", null, null, null, null,
null);
		FileAppender file = FileAppender.createAppender("./logs/file/promatic.log", "false",
"false", "file", "false", "true", "false", null, layout, null, "false", null, null);
		logger.addAppender(file);

		// Log Level: DEBUG
		logger.setLevel(Level.DEBUG);

		// After: Logs (promatic.log) above DEBUG Level
		logger.debug("After: [DEBUG] Test log4j 2.");
```

```
        logger.info("After: [INFO] Test log4j 2.");
        logger.warn("After: [WARN] Test log4j 2.");
        logger.error("After: [ERROR] Test log4j 2.");
        logger.fatal("After: [FATAL] Test log4j 2.");

        /*
        Logs
        ===================
        After: [DEBUG] Test log4j 2.
        After: [INFO] Test log4j 2.
        After: [WARN] Test log4j 2.
        After: [ERROR] Test log4j 2.
        After: [FATAL] Test log4j 2.
        */

}
```

References

Apache Log4j Core 2.0-rc1 API