Logging using SLF4J HandsOn

Exercise 1: Logging Error Messages and Warning Levels

Task: Write a Java application that demonstrates logging error messages and warning levels using SLF4J.

Code:

LoggingExample.java

**package** SLF4JExamplecode;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** LoggingExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(LoggingExample.**class**);

**public** **static** **void** main(String[] args) {

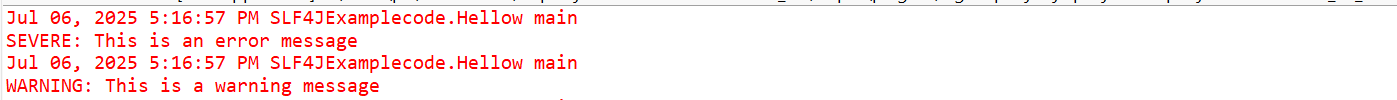
***logger***.error("This is an error message");

***logger***.warn("This is a warning message");

}

}

Output:



Exercise 2: Parameterized Logging

Task: Write a Java application that demonstrates parameterized logging using SLF4J.

Code:

**package** SLF4JExamplecode;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** ParameterizedLoggingExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(ParameterizedLoggingExample.**class**);

**public** **static** **void** main(String[] args) {

String username = "john\_doe";

**int** age = 25;

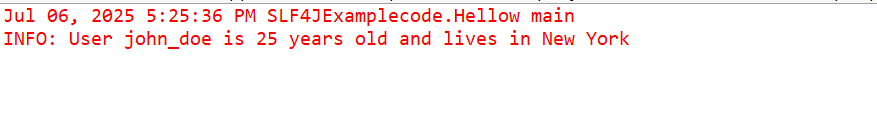
String city = "New York";

// Parameterized logging - efficient and clean

***logger***.info("User {} is {} years old and lives in {}", username, age, city);

}

Output:



Exercise 3: Using Different Appenders

Task: Write a Java application that demonstrates using different appenders with SLF4J.

Code:

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** AppendersExample {

**private** **static** **final** Logger ***logger*** = LoggerFactory.*getLogger*(AppendersExample.**class**);

**public** **static** **void** main(String[] args) {

// These messages will go to both console and file (app.log)

***logger***.info("Application started");

***logger***.warn("This is a warning message");

***logger***.error("This is an error message");

***logger***.debug("Debug message - might not appear depending on log level");

// Simulate some application work

**for** (**int** i = 1; i <= 3; i++) {

***logger***.info("Processing item {}", i);

}

***logger***.info("Application finished");

}

}}

Output:

