## 

Mubeen Khalid

(SP21-BSE-015)

Design Patterns Lab

Chain of Responsibility Pattern

Updated Code:

AbstractLogger.java:

package UpdatedCode;

abstract class AbstractLogger {

    public static int INFO = 1;

    public static int DEBUG = 2;

    public static int ERROR = 3;

    public static int WARNING = 4;

    protected int level;

    // next element in chain or responsibility

    protected AbstractLogger nextLogger;

    public void setNextLogger(AbstractLogger nextLogger) {

        this.nextLogger = nextLogger;

    }

    public void logMessage(int level, String message) {

        if (this.level == level) {

            write(message);

        } else if (nextLogger != null) {

            nextLogger.logMessage(level, message);

        }

    }

    abstract protected void write(String message);

}

ConsoleLogger.java:

package UpdatedCode;

class ConsoleLogger extends AbstractLogger {

    public ConsoleLogger(int level) {

        this.level = level;

    }

    @Override

    protected void write(String message) {

        System.out.println("Standard Console::Logger: " + message);

    }

}

DatabaseLogger.java:

package UpdatedCode;

class DatabaseLogger extends AbstractLogger {

    public DatabaseLogger(int level){

       this.level = level;

    }

    @Override

    protected void write(String message) {

       System.out.println("Database::Logger: " + message);

    }

 }

FileLogger.java:

package UpdatedCode;

class FileLogger extends AbstractLogger {

    public FileLogger(int level){

       this.level = level;

    }

    @Override

    protected void write(String message) {

       System.out.println("File::Logger: " + message);

    }

 }

ErrorLogger.java:

package UpdatedCode;

class ErrorLogger extends AbstractLogger {

    public ErrorLogger(int level){

       this.level = level;

    }

    @Override

    protected void write(String message) {

       System.out.println("Error Console::Logger: " + message);

    }

 }

ChainPatternDemo.java:

package UpdatedCode;

public class ChainPatternDemo {

    private static AbstractLogger getChainOfLoggers() {

        AbstractLogger errorLogger = new ErrorLogger(AbstractLogger.ERROR);

        AbstractLogger fileLogger = new FileLogger(AbstractLogger.DEBUG);

        AbstractLogger databaseLogger = new DatabaseLogger(AbstractLogger.WARNING);

        AbstractLogger consoleLogger = new ConsoleLogger(AbstractLogger.INFO);

        errorLogger.setNextLogger(databaseLogger);

        databaseLogger.setNextLogger(fileLogger);

        fileLogger.setNextLogger(consoleLogger);

        return errorLogger;

    }

    public static void main(String[] args) {

        AbstractLogger loggerChain = getChainOfLoggers();

        loggerChain.logMessage(AbstractLogger.INFO, "This is an information.");

        loggerChain.logMessage(AbstractLogger.DEBUG, "This is a debug level information.");

        loggerChain.logMessage(AbstractLogger.WARNING, "This is a warning information.");

        loggerChain.logMessage(AbstractLogger.ERROR, "This is an error information.");

    }

}

Output:

