# Singleton Pattern for Logging Utility

1. **What is the Singleton Pattern**

## Simple Explanation

The Singleton Pattern makes sure a class has only one object in the whole pro- gram. For a logger, this means every part of the app uses the same logger to print messages, keeping all logs consistent, like writing to one place.

# Setting Up the Logger

## Logger Class Details

The Logger class is built to be a Singleton with:

* + - A private static variable called myLogger to store the one Logger object.
    - A private constructor so no one can make a new Logger.
    - A public static method, getLogger, to give the Logger object, creating it if it doesn’t exist yet.

# Implementing the Singleton

## How the Logger Works

The Logger class follows the Singleton Pattern like this:

* + - myLogger holds the single Logger object.
    - The private constructor stops anyone from creating extra Loggers.
    - getLogger checks if myLogger is null and makes a new Logger if needed, then returns it.
    - The print method shows messages on the screen.

# Testing the Singleton

## Checking the Singleton

The LoggerTest class tests if the Singleton works:

* + - Gets two Loggers using getLogger.
    - Prints a message with each Logger.
    - Checks if both Loggers are the same using ==. Running LoggerTest gives this output:

Message: Hello! Message: Hi again! It’s the same logger!

This shows there’s only one Logger, so the Singleton works.