```
using System;
using System.Configuration;

namespace ConsoleApplication1
{
    public class JobLogger
    {
        private static bool _logToFile;
        private static bool _logToConsole;
        private static bool _logMessage;
        private static bool _logWarning;
        private static bool _logError;
        private static bool _logToDatabase;

        //  I have cut off this line "private bool _initialized;" since it was
unused on the code.


        public JobLogger(bool logToFile, bool logToConsole, bool logToDatabase, bool
logMessage, bool logWarning, bool logError)
        {
            _logError = logError;
            _logMessage = logMessage;
            _logWarning = logWarning;
            _logToDatabase = logToDatabase;
            _logToFile = logToFile;
            _logToConsole = logToConsole;
        }
        public static void LogMessage(string message, bool message_, bool warning,
bool error)
            // Not recommend to use same variable names (message) even if the
variable types are different, let's avoid confusion.
        {
            message.Trim();
            if (string.IsNullOrEmpty(message))
            {
                return;
            }
            if (!_logToConsole && !_logToFile && !_logToDatabase)
            {
                throw new Exception("Invalid configuration");
            }

            if ((!_logError && !_logMessage && !_logWarning) || (!message_ &&
!warning && !error))
            {
                throw new Exception("Error or Warning or Message must be
specified");
            }

            //  The variable needed to be referenced.
            int t = 0;
            if (message_ && _logMessage)
            {
                t = 1;
            }
            if (error && _logError)
            {
```

```
                t = 2;
            }
            if (warning && _logWarning)
            {
                t = 3;
            }

            //  Adding the "Using" statement for best practices and the sql
connection will automatically close by itself.
            //  And also added "try/catch" for best practises whenever any exception
is being triggered.
            using (System.Data.SqlClient.SqlConnection connection = new
System.Data.SqlClient.SqlConnection(ConfigurationManager.AppSettings["ConnectionStri
ng"]))
            {
                try
                {
                    connection.Open();
                    System.Data.SqlClient.SqlCommand command = new
                    System.Data.SqlClient.SqlCommand("Insert into Log Values('" +
message + "', " + t + ")");
                    command.ExecuteNonQuery();
                }
                catch (Exception)
                {
                    throw;
                }
            }

            //  The variable needed to be referenced.
            var l = string.Empty;

            //  Since the same sentence was called on the three different "if
validations", I have put the three into one validation to only makes one call.
            if ((error && _logError) || (warning && _logWarning) || (message_ &&
_logMessage))
            {
                l = l + DateTime.Now.ToShortDateString() + message;
            }

            //  I have reordered the call when validating the "if statement" since
it was making unnecessary calls. I also added "else".
            if

(!System.IO.File.Exists(ConfigurationManager.AppSettings["LogFileDirectory"] +
"LogFile" + DateTime.Now.ToShortDateString() + ".txt"))
            {

System.IO.File.WriteAllText(ConfigurationManager.AppSettings["LogFileDirectory"] +
"LogFile" + DateTime.Now.ToShortDateString() + ".txt", l);
            }
            else
            {
                l =
System.IO.File.ReadAllText(ConfigurationManager.AppSettings["LogFileDirectory"] +
"LogFile" + DateTime.Now.ToShortDateString() + ".txt");
            }
```

```
            if (error && _logError)
            {
                Console.ForegroundColor = ConsoleColor.Red;
            }
            if (warning && _logWarning)
            {
                Console.ForegroundColor = ConsoleColor.Yellow;
            }
            if (message_ && _logMessage)
            {
                Console.ForegroundColor = ConsoleColor.White;
            }
            Console.WriteLine(DateTime.Now.ToShortDateString() + message);
        }

    }
}
```