using System;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using System.Configuration;

using System.IO;

namespace Test

{

public class JobLogger

{

private static bool \_logToFile;

private static bool \_logToConsole;

private static bool \_logToDatabase;

private static bool \_logMessage;

private static bool \_logWarning;

private static bool \_logError;

public JobLogger(bool logToFile, bool logToConsole, bool logToDatabase, bool

logMessage, bool logWarning, bool logError)

{

\_logToFile = logToFile;

\_logToConsole = logToConsole;

\_logToDatabase = logToDatabase;

\_logMessage = logMessage;

\_logWarning = logWarning;

\_logError = logError;

}

public static void LogMessage(string message, EnumeratorClass.LogType pLogType)

{

if (message == null || message.Trim().Length == 0)

{

return;

}

if (!\_logToConsole && !\_logToFile && !\_logToDatabase)

{

throw new Exception("Invalid configuration");

}

if (!\_logError && !\_logMessage && !\_logWarning)

{

throw new Exception("Error or Warning or Message must be specified");

}

if((pLogType == EnumeratorClass.LogType.message && !\_logMessage) ||

(pLogType == EnumeratorClass.LogType.error && !\_logError) ||

(pLogType == EnumeratorClass.LogType.warning && !\_logWarning))

{

throw new Exception("JobLogger is not configured to save this type of log");

}

if (\_logToDatabase)

{

SqlConnection connection = new SqlConnection(ConfigurationManager.AppSettings["ConnectionString"]);

SqlCommand command = new SqlCommand(string.Format("Insert into Log Values('{0}',{1}", message, pLogType.GetHashCode().ToString()), connection);

try

{

connection.Open();

command.ExecuteNonQuery();

}

catch (Exception ex)

{

throw ex;

}

finally

{

connection.Close();

connection.Dispose();

command.Dispose();

}

}

if (\_logToFile)

{

try

{

File.AppendAllText(string.Format(@"{0}\LogFile{1}.txt", ConfigurationManager.AppSettings["LogFileDirectory"], DateTime.Now.ToString("yyyy-MM-dd")), DateTime.Now.ToShortDateString() + message);

}

catch(Exception)

{

throw new Exception("Exception when trying to write to log file");

}

}

if (\_logToConsole)

{

switch (pLogType)

{

case EnumeratorClass.LogType.message:

{

Console.ForegroundColor = ConsoleColor.White;

break;

}

case EnumeratorClass.LogType.error:

{

Console.ForegroundColor = ConsoleColor.Red;

break;

}

case EnumeratorClass.LogType.warning:

{

Console.ForegroundColor = ConsoleColor.Yellow;

break;

}

}

Console.WriteLine(string.Format("{0} {1}", DateTime.Now.ToShortDateString(), message));

}

}

}

public class EnumeratorClass

{

public enum LogType

{

message = 1,

error = 2,

warning = 3

}

}

}