**Class name: SyncSharp.Storage.Logger**

Scope / Modifier: public static (lifespan: Application-wide)

Description:

Throughout the duration period of the runtime for a particular synctask, Logger handles:

1. Log recording operations from
   1. Sync preview plan summary
   2. File or folder I/O operations.
   3. Sync results summary
   4. System thrown exceptions (during current sync session).
2. Deletion of log file
3. Reading of log file contents

Directory location of stored log file:

{root path of Syncsharp.exe}/Profiles/{Machine ID}/{Sync task name}/

***Attributes:***

**public enum LogType { CopySRC, CopyTGT, DeleteSRC, DeleteTGT, RenameSRC, RenameTGT, CreateSRC, CreateTGT };**

*Description: Enumerable log types to differentiate type of log operation to be carried out.*

*Enum types:*

*CopySRC:- File copy operation that originates from source location.*

*CopyTGT:- File copy operation that originates from target location.*

*DeleteSRC:- File copy operation that originates from source location.*

*DeleteTGT:- File copy operation that originates from target location.*

*RenameSRC:- File copy operation that originates from source location.*

*RenameTGT:- File copy operation that originates from target location.*

*CreateSRC:- Folder create operation that originates from source location.*

*CreateTGT:- Folder create operation that originates from target location.*

***general method call usage flow***

**WriteSyncLog** () //start syncTask  
  
**WritePreviewLog** () //log finalised preview plan  
  
//log file operations  
**WriteLog**()  
**WriteLog**()  
**WriteLog**()  
  
**WriteSyncLog** () //end syncTask

***Method operations:***

1a) Log recording operations from initiation of Sync & completed Sync results summary

**public static bool WriteSyncLog(string metaDataDir, string syncTaskName, bool start)**

*Action: Set begin or end (****to signal start and completion****) of a perticular synctask log recording*

*Logic:*

   if (start)  
                                        WriteLogHeader();  
                                else  
                                        WriteSyncResultsLogEntry();

Begin: **WriteLogHeader()**

*Actions:*

* *Append sync begin log header to log file for current sync task session.*
* *Append record timestamp.*
* *Reset file and folder operation counters.*

End: **WriteSyncResultsLogEntry()**

*Actions:*

* + - * + *Append* file & folder I/O operation results count summary
        + *Record timestamp.*
        + *Append* log footer for current synctask.

*Parameters:*

*metaDataDir:- path of metadata directory*

*[{root path of Syncsharp.exe}/Profiles/{Machine ID}/]*

*syncTaskName:- name of sync task*

start*:- begin (if true), end (if false)*

*Return: success state*

1b) Log recording operations from Sync preview plan summary

**public static bool WritePreviewLog(string metaDataDir, string syncTaskName, int srcCopyTotal, long srcCopySize, int srcDeleteTotal, long srcDeleteSize, int srcRenameTotal, long srcRenameSize, int tgtCopyTotal, long tgtCopySize, int tgtDeleteTotal, long tgtDeleteSize, int tgtRenameTotal, long tgtRenameSize) {}**

*Action:* Record file I/O operation count summary to be carried out for sync preview for current sync task session into log file

*Parameters:*

*metaDataDir:- path of metadata directory*

*[{root path of Syncsharp.exe}/Profiles/{Machine ID}/]*

*syncTaskName:- name of sync task*

*srcCopyTotal:- total number of files to copy from source location.  
        srcCopySize:- total filesize(bytes) of files to copy from source location.*

*srcDeleteTotal:- total number of files to delete from source location.  
        srcDeleteSize:- total filesize(bytes) of files to delete from source location.*

*srcRenameTotal:- total number of files to rename from source location.  
        srcRenameSize:- total filesize(bytes) of files to rename from source location.*

*tgtCopyTotal:- total number of files to copy from target location.  
        tgtCopySize:- total filesize(bytes) of files to copy from target location.*

*tgtDeleteTotal:- total number of files to delete from target location.  
        tgtDeleteSize:- total filesize(bytes) of files to delete from target location.*

*tgtRenameTotal:- total number of files to rename from target location.  
        tgtRenameSize:- total filesize(bytes) of files to rename from target location.*

*Return: success state*

1c) Log recording operations from File or folder I/O operations.

**public static bool WriteLog(LogType logType, string srcPath, long srcSize, string tgtPath, long tgtSize)**

*Action:* Record file I/O operation count summary to be carried out for sync preview for current sync task session into log file

*Parameters:*

logType*:- name of sync task*

srcPath*:- file* path on source *path*.

srcSize*:-* filesize*(bytes)* on source*.*

tgtPath*:- total number of files to copy from target path.*tgtSize*:-* filesize*(bytes)* on target*.*

*Expected parameters when calling for particular logType:*

*CopySRC: requires file path and size to copy from source and to destination*

**WriteLog(LogType***.CopySRC***, srcPath, srcSize, tgtPath, tgtSize);**

*CopyTGT: requires file path and size to copy from destination and to source*

**WriteLog(LogType***.CopyTGT***, srcPath, srcSize, tgtPath, tgtSize);**

*RenameSRC: requires file path and size to rename from source and to destination*

**WriteLog(LogType***.RenameSRC***, srcPath, srcSize, tgtPath, tgtSize);**

*RenameTGT: requires file path and size to rename from destination and to source*

**WriteLog(LogType***.RenameTGT***, srcPath, srcSize, tgtPath, tgtSize);**

*DeleteSRC: requires file path and size to delete from source*

**WriteLog(LogType***.DeleteSRC***, srcPath, srcSize, null, 0);**

*DeleteTGT: requires file path and size to delete from destination*

**WriteLog(LogType***.DeleteTGT***, null, 0, tgtPath, tgtSize);**

*CreateSRC: requires folder path and size to create on source*

**WriteLog(LogType***.CreateSRC***, srcPath, 0, null, 0);**

*CreateTGT: requires folder path and size to create on destination*

**WriteLog(LogType***.CreateTGT***, null, 0, null, tgtSize);**

*Note: null or 0 values passed in as input parameters have no relevance for the particular selected logtype in use.* For these unused parameters for specific operations, caller passes in *-Path =* ***null****; -Size =* ***0****;*

*Return: success state*

1d) Log recording operations from System thrown exceptions (during current sync session).

**public static bool WriteErrorLog(string errorMsg) {}**

*Action: reports general program error exceptions thrown into log file*

*Parameters:*

*errorMsg:- system error description*

*Return: success state*

2) Deletion of log file

**public static bool DeleteLog(string syncTaskName)**

*Action: Delete log file and return whether successful.*

*Parameters:  
        syncTaskName:- name of sync task*

*Return: success state*

3) Reading of log file contents

**public static string ReadLog(string syncTaskName)**

*Action: Read log file and return log contents to caller.*

*Parameters:*

*syncTaskName:- name of sync task*

*Return: log contents*