

# Saves a file

/\*

Copyright (c) 2010 [James Craig](#)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.\*/

```
#region Usings
using System;
```

```
using System.Collections.Generic;
using System.IO;
using System.Net;
using System.Text;

#endregion

namespace Utilities
{
    ///
    /// Utility class for managing files
    ///
    public static class FileManager
    {
        ///
        /// Determines if a directory exists
        ///
        /// Path of the directory /// true if it exists, false
        otherwise
        public static bool DirectoryExists(string DirectoryPath)
        {
            try
            {
                return Directory.Exists(DirectoryPath);
            }
            catch (Exception a)
            {
                throw a;
            }
        }
        ///
        /// Saves a file
        ///
        /// Content of the file /// Path of the file /// Tells the
        system if you wish to append data or create a new document
        public static void SaveFile(string Content, string
```

```
FileName, bool Append)
{
FileStream Writer = null;
try
{
byte[] ContentBytes = Encoding.UTF8.GetBytes(Content);
int Index = FileName.LastIndexOf('/');
if (Index <= 0) { Index = FileName.LastIndexOf('.'); } if
(Index <= 0) { throw new Exception("Directory must be
specified for the file"); } string Directory =
FileName.Remove(Index) + "/"; bool Opened = false; while
(!Opened) { try { if (Append) { Writer = File.Open(FileName,
FileMode.Append, FileAccess.Write, FileShare.None); } else {
Writer = File.Open(FileName, FileMode.Create,
FileAccess.Write, FileShare.None); } Opened = true; } catch
(System.IO.IOException e) { throw e; } }
Writer.Write(ContentBytes, 0, ContentBytes.Length);
Writer.Close(); } catch (Exception a) { throw a; } finally {
if (Writer != null) { Writer.Close(); Writer.Dispose(); } } }
} } [/csharp]
```