Clean a directory without deleting it

```
#region License and Copyright
/* ————————————————————————-
* Dotnet Commons IO
*
*
* This library is free software; you can redistribute it and/or modify it
* under the terms of the GNU Lesser General Public License as published by
* the Free Software Foundation; either version 2.1 of the License, or
* (at your option) any later version.
*
* This library is distributed in the hope that it will be useful, but
* WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY
* or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License
* for more details.
*
* You should have received a copy of the GNU Lesser General Public License
* along with this library; if not, write to the
*
* Free Software Foundation, Inc.,
* 59 Temple Place,
* Suite 330,
* Boston,
* MA 02111-1307
* USA
```

```
*
* ---------------------------
*/
#endregion

using System;
using System.Collections;
using System.Globalization;
using System.IO;

namespace Dotnet.Commons.IO
{
class MainClass{
/// ----------------------------
///

Clean a directory without deleting it.
/// directory path to clean /// in the case when cleaning is
unsuccessful
/// ----------------------------
public static void CleanDirectory(string path)
{
bool tmpBool;
if (File.Exists(path))
tmpBool = true;
else
tmpBool = Directory.Exists(path);

if (!tmpBool)
throw new ArgumentException(path + " does not exist");

if (!Directory.Exists(path))
throw new ArgumentException(path + " is not a directory");

IOException exception = null;

DirectoryInfo dir = new DirectoryInfo(path);
FileInfo[] files = dir.GetFiles();
foreach(FileInfo f in files)
```

```
{
try
{
f.Delete();
}
catch (IOException ioe)
{
exception = ioe;
}
}
```

```
DirectoryInfo[] subdirs = dir.GetDirectories();
for (int i = 0; i < subdirs.Length; i++) { try { subdirs[i].Delete(true); } catch (IOException ioe) { exception = ioe; } } if (null != exception) { throw exception; } } } }
```

[/csharp]