using ConsoleApp2.FileIO;

using System;

using System.Collections.Generic;

using System.Diagnostics.Contracts;

using System.IO;

using System.Linq;

using System.Reflection;

using System.Security;

using System.Security.Permissions;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp2

{

public class Program

{

static void Main(string[] args)

{

//string path = AppDomain.CurrentDomain.SetupInformation.ApplicationBase;

//Console.WriteLine(path);

var sandBox=Helper.CreateSandBox();

var type = typeof(Utils);

var utils = sandBox.CreateInstanceAndUnwrap(type.Assembly.FullName, type.FullName) as Utils;

var path = Environment.GetFolderPath(Environment.SpecialFolder.Desktop) + @"\New\a.txt";

Console.WriteLine(utils.GetFileText(path));

AppDomain.Unload(sandBox);

}

}

public static class Helper

{

public static AppDomain CreateSandBox()

{

Contract.Ensures(Contract.Result<AppDomain>() != null);

var platform = Assembly.GetExecutingAssembly();

var name=platform.FullName+": Sandbox"+Guid.NewGuid();

var setup = new AppDomainSetup { ApplicationBase = Path.GetDirectoryName(platform.Location) };

PermissionSet permissionSet = new PermissionSet(PermissionState.None);

string testFolder = Environment.GetFolderPath(Environment.SpecialFolder.Desktop) + @"\New";

permissionSet.AddPermission(new FileIOPermission(FileIOPermissionAccess.Read, testFolder));

permissionSet.AddPermission(new SecurityPermission(SecurityPermissionFlag.Execution));

var sandbox=AppDomain.CreateDomain(name,null,setup,permissionSet);

Contract.Assume(sandbox!=null);

return sandbox;

}

}

namespace FileIO

{

public class Utils : MarshalByRefObject

{

public string GetFileText(string textFileName)

{

return File.ReadAllText(textFileName);

}

}

}

}