[1. .NET Framework 版本 2](#_Toc464819030)

[1-1. Win 10 X64 系统自带两套X86 和X64的framework 2](#_Toc464819031)

[2. CLR 和 JIT 都在什么位置 3](#_Toc464819032)

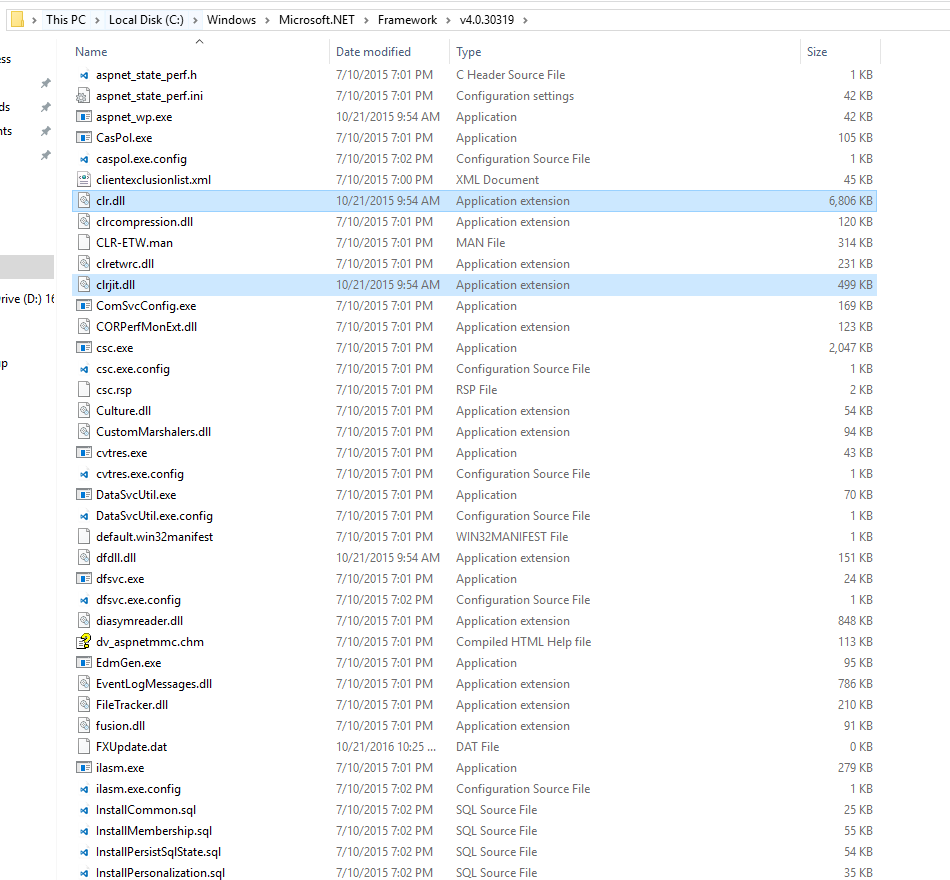
[2-1. CLR是如何被加载的 4](#_Toc464819033)

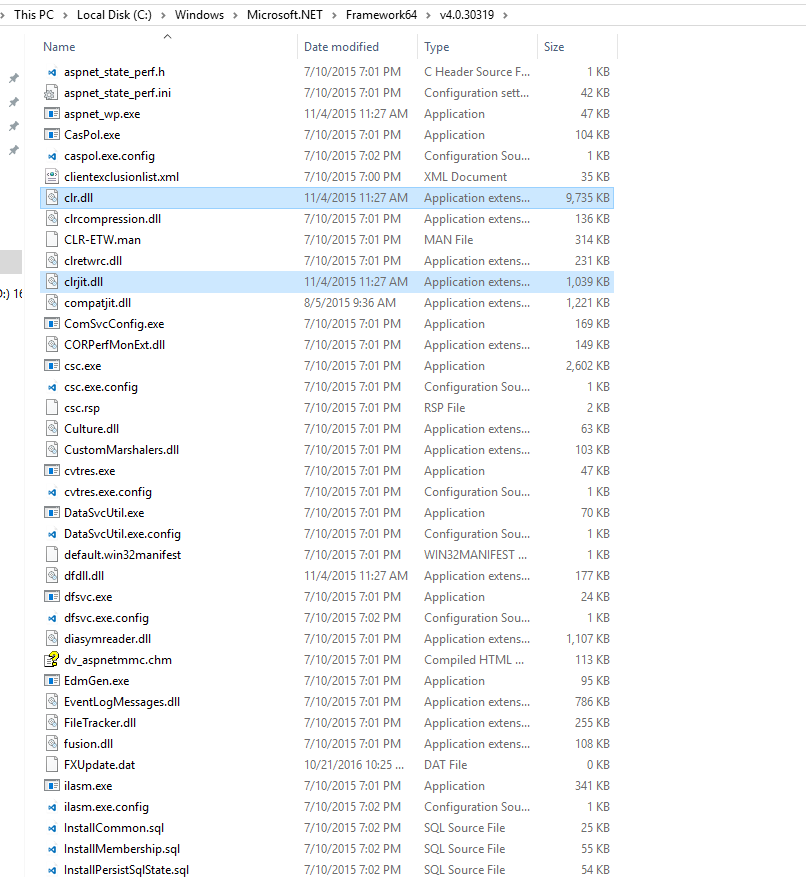
[3. 所有的CLR 类库文件不仅在GAC里有，CLR目录下都有 6](#_Toc464819034)

# .NET Framework 版本

## Win 10 X64 系统自带两套X86 和X64的framework

一个在Framework 目录下，一个在Framework64目录下



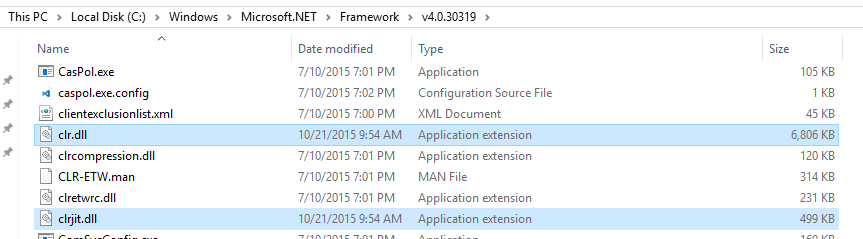


# CLR 和 JIT 都在什么位置

<http://stackoverflow.com/questions/30802270/where-exactly-is-net-runtime-clr-jit-compiler-located>

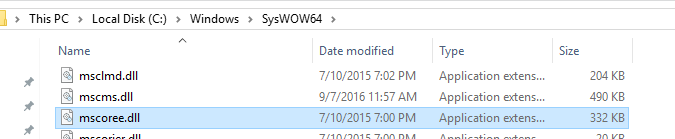
C:\Windows\Microsoft.NET\Framework\v4.0.30319\clrjit.dll. The x64 version is in the Framework64 directory. The .NET v2 version had a different name, mscorjit.dll, find it back in the v2.0.50727 directories.

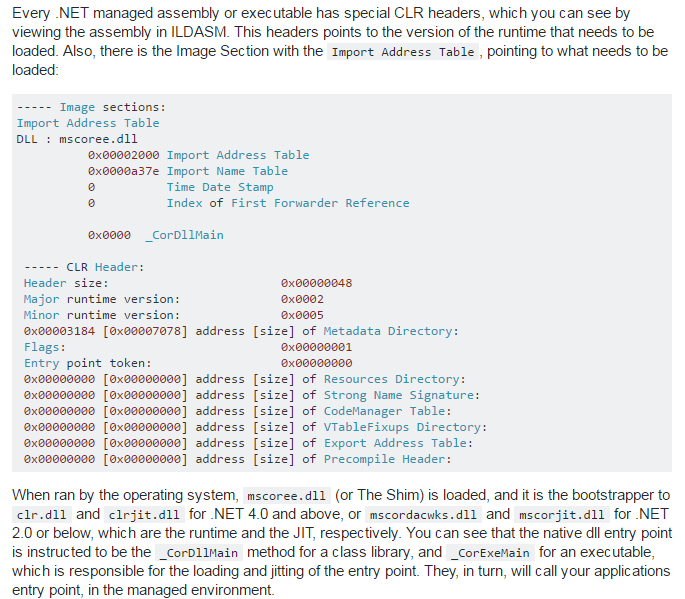
It is clr.dll in the v4 versions, mscorwks.dll and mscorsvc.dll in the v2 versions. Two different ones back then with different garbage collectors, "wks" is the workstation version, "svc" is the server version



## CLR是如何被加载的

"how does the CLR get loaded?" That's the job of c:\windows\syswow64\mscoree.dll, you'll use c:\windows\system32\mscoree.dll when you target x64 in your EXE project. Every .NET assembly has 5 or 9 bytes of unmanaged code, a jump into that DLL. Either \_CorExeMain or \_CorDllMain, depending on whether the assembly was built as an exe or a library. mscoree.dll takes a look at the metadata in the assembly and decides what version of the CLR needs to loaded so it can be properly executed.





# 所有的CLR 类库文件不仅在GAC里有，CLR目录下都有

