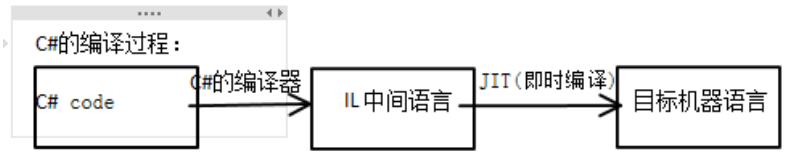
CSC.exe编译Visual C#的代码文件

C#的编译过程



如何用CSC.exe来编译Visual C#的代码文件

- Csc.exe 编译器的位置路径：C:\Windows\Microsoft.NET\Framework64\v4.0.30319
- 如何编译：

1.为了使用方便，你可以手动把上面的目录添加到Path环境变量中去

2.用Csc.exe编译HelloWorld.cs非常简单，打开命令提示符，并切换到存放 HelloWorld.cs文件的目录中，输入下列行命令:

Csc /r:system.dll HelloWorld.cs

命令选项：

#	Option	Remark
1	@	这个选项是用来指定响应文件。响应文件是一种包含了许多编译选项的文件。这些编译选项将和源代码文件一起由编译器进行处理。一般来说此种响应文件是以文本文件形式出现。他的扩展名.rsp。在响应文件中是用＃符号表示开始的注释。 例：以下是一个响应文件resp1.rsp的内容： # 这是一个简单的响应文件，文件名称为resp1.rsp #使用方法: csc @resp1.rsp /target:exe /out:sample.exe sample.cs 此响应文件的作用就是把sample.cs文件编译成sample.exe文件。如果在一次编译中要指定多个响应文件，可以指定多个响应文件选项，如：@file1.rsp @file2.rsp
2	/? 与 /help	
3	/addmodule	本选项是使编译器搜集从用户正在编译的工程到可用文件中所有类型的信息。所有添加了/addmodule的模块在运行时必须与输出文件在同一目录中。这就是说，用户可以在编译时指定任何目录中的模块，但在运行时这个模块必须在应用程序目录中。文件中不能包含汇编名单。例如：如果输出文件用/taarget:module创建，其元数据可以用/addmodule导入。 例子：把二个模块加入myProject.cs中 csc /addmodule:module1.dll;module2.dll myProject.cs

4	/baseaddress	本选项允许用户指定载入DLL时的首选地址，这个首选地址可以是十进制、十六进制、八进制。DLL的缺省首选地址在.Net运行时设置。如果目标文件不是DLL文件，这个选项将被忽略。 例子：把myLibrary.cs编译成DLL文件，并且当此DLL在.Net运行环境被载入时的地址是0x1111000：csc /baseaddres:0x1111000 /target:library myLibrary.cs
5	/bugreport	这个选项用来报告编译时的错误信息。在报告中包含以下内容： 1）.编译中所有源代码的一个拷贝 2）.在编译中所有的编译选项 3）.编译信息，包括编译器、运行时间、操作系统的版本信息 4）.编译器输出 5）.问题的描述 6）.如何解决问题的描述 例子：生成一个bugs.txt文件，并把错误报告放在文件里面 csc /bugreport:bugs.txt Hello.cs
6	/checked	此选项指定不在检验或或者未检验关键字范围内以及导致超出数据类型范围的值的整数计算语句是否产生运行例外。具体的说就是，如果不在检验或者未检验关键字范围内的整数计算语句产生的值在数据类型允许的范围之外，并且在编译中使用了/checked+(/checked)，该语句就会在运行时产生例外，如果在编译时使用了/checked-，在运行时该语句就不会产生例外。 例子：编译myMath.cs，并且指定一个不在检验或者未检验关键字范围内的整数计算语句（且其产生的值超出数据类型的范围），将在运行时引起例外。 csc /checked+ myMath.cs
7	/codepage	如果用户编译的一个或者多个源代码不使用计算机上的默认代码页，可以使用/codepage选项来指定希望使用的代码页。/codepage适用于编译中所有的源代码文件。 如果源代码文件在计算机上的同一个代码页位置创建，或者源代码文件用UNICODE或者UTF-8来创建，用户就不需要使用/codepage了。
8	/debug	此选项是在调试时候使用的，当调试者启用了这个选项来调试自己的程序，将会创建一个.pdb文件，并把各种调试信息写到此文件里。有两选项来指定调试的类型： • /debug [+/-] :当选用/debug +就会把创建.pdb文件，并把调试信息存储到里面；/debug -是一个缺省设置，就是不产生任何调试信息。 • /debug:[full/pdbonly] :当使用/debug:full就是创建缺省的调试信息，有点类似/debug+选项。/debug: pdbonly选项是创建.pdb文件，并且你只能使用源代码调试在调试工具里。 例子：编译Hello.cs并且为Hello.cs创建调试信息 csc /debug+ HelloWorld.cs
9	/define	此选项在程序中定义了一个符号，他和在源程序中使用#define预处理程序指示功能相同，此符号保持已定义状态，直到源文件中的#undefine指示符删除定义或者编译器已到达了文件末尾。你可以用/d简写来代替。 例子：下面是my.cs的源程序 using System;

		public class myBuild
		{
		public static void Main()
		{
		#if (final)
		Console.WriteLine("Final Build");
		Console.ReadLine();
		#else
		Console.WriteLine("Trial Build");
		Console.ReadLine();
		#endif
		
		}
		}
		如果用csc /define:final my.cs来编译就会显示"Final Build"，如果没有/define，编译后执行就会显示"Trial Build"。

10	/doc	文档在当今已经变得愈来愈重要了，一个好的程序应该配有相当的文档。如果你在写程序的文档中用的是"///"标识符来注释。当你使用/doc选项来编译时，你的所以注释文档将会自动的保留在一个XML文件中。
		例子：以下是my.cs 的源程序
		using System;
		/// <summary>
		/// This is a sample class that has some documentation
		/// </summary>
		public class myBuild
		{
		/// <summary>
		/// Main entry point of the class
		/// </summary>
		public static void Main()
		{
		}
		}
		用下列编译语句会产生my.xml文件，看看my.xml文件到底存储了什么东西。
		Csc /doc:my.xml my.cs

11	/fullpaths	在默认情况下，编译产生的错误或者警告都只会指明发现错误的文件名称，加入此选项使得在编译器产生错误或者警告的时候会显示完整的路径。你可以把上面的my.cs程序语法搞错，再用 csc /fullpaths my.cs 和 csc my.cs分别编译，看看错误提示有什么不同。

12	/incremental	本选项主要是激活增量编译器，这种编译器只对上次编译后发生改变的函数进行编译。如果在编译时候选用了/debug选项，调试信息的状态存储在相应的.pdb文件中。除此之外编译时的信息都存储在.incr文件中，此.incr文件的名称为output_file_name.extension.incr。即如果输出文件时out.exe，则此文件对应的incr文件是out.exe.incr文件。 例子：利用增量编译器来编译文件 csc /incremental /out:my.exe my.cs 如果编译成功则会产生2个文件，分别是：my.exe和my.exe.incr。
13	/linkresource	这个选项就是在输出文件中创建到.Net资源的链接。他的简写是/linkres。资源文件就是在那些在工程文件中使用到的所有的资源，像图片、声音等。这个选项只是对于资源文件建立链接，这样有助于管理使用同一资源的程序，而不需要多个副本。此选项的具体语法如下： /linkresource:filename,identifier,mimetype 其中： filename:是想建立链接的.Net的资源文件 identifier(可选)：资源的逻辑名称，该名称用于载入资源，默认名称是文件名称。 mimetype(可选)：是一个代表资源的媒介类型的字符串。默认为空。 例子:在文件中建立一个指向reso.resource的链接 csc /linkres:reso.resource myResource.cs
14	/main	当我们编译二个或者多个拥有Main方法的Class，我们可以使用这个选项让用户指定最终的输出文件中的使用那个Main的方法。 例子：编译二个文件，但输出文件中的Main方法来自Main1 Class csc myMain1.cs myMain2.cs /main:Main1
15	/nologo	这个选项禁止在编译器启动时显示开始标志和编译过程中显示报告信息。 例子：csc /nologo my.cs
16	/nooutput	编译文件，但不创建任何输出文件。用户可以看到任何编译错误和警告。 例子：csc /nooutput my.cs
17	/nostdlib	这个选项禁止导入mscorlib.dll。这个DLL包含了这个系统名称空间。当用户希望使用自己的系统名称空间时，一般才会使用此选项。 例子：编译文件，但不导入mscorlib.dl csc /nooutput myOutput.cs
18	/nowarn	选项是在编译过程中禁止指定的警告类型。如果是禁止多个警告类型，用逗号分隔。 例子：在编译过程中禁止警告类型CS0108和CS0109 csc /nowarn:108,109 Warn.cs
19	/optimize	本选项激活或者禁用由编译器执行优化。优化的结果是使得输出文件更小、更快、更有效率。缺省是/optimize执行优化，如果你选用了/optimize-则禁止优化。/o是/optimize的简写。 例子:编译文件，并禁止优化 csc /optimise- my.cs

20	/out	在没有指定输出文件的情况下，如果通过编译器编译后文件是EXE文件，则输出文件将从包含Main方法的源代码的文件中获得名字；如果编译后的文件是DLL文件，将从第一个源代码文件中获得名字。如果用户想要指定输出文件名称，就可以使用此选项。 例子：编译HelloWord.cs文件，并把输出文件命名为Hello.exe csc /out:Hello.exe helloworld.cs
21	/recurse	此选项允许用户编译在指定目录或者工程目录的所有子目录中的所有源代码文件。用户可以使用通配符来编译工程目录下的所有匹配文件。 例子：编译/dir1/dir2目录下及其下级目录中的所有C#文件，并生成dir2.dll csc /target:library /out:dir2.dll /recurse: dir1\dir2*.cs
22	/refrence	此选项可使得当前编译工程使用指定文件中的公共类型信息。这个选项对于初学者是很重要的。此选项的简写是/r。你必须引用在程序代码中使用"using"关键字导入的所有文件，如果在你的程序中，使用了自己编写的类库，在编译时也必须引用。 例子：编译文件，并引用在程序中使用的文件 csc /r:system.dll;myExec.exe;myLibrary.dll myProject.cs (注：其中那个myExec.exe和myLibrary.dll时自己创建的)
23	/target	这个选项是告诉编译器你所想得到什么类型的输出文件。除非使用/target:module选项，其他选项创建的输出文件都包含着汇编名单。汇编名单存储着编译中所有文件的信息。在一个命令行中如果生成多个输出文件，但只创建一个汇编名单，并存储在第一个输出文件中。 下面是/target的4种用法： /target:exe 创建一个可执行（EXE）的控制台应用程序 /target:library 创建一个代码库（DLL） /target:winexe 创建一个windows程序（EXE） /target:module 创建一个模块（DLL） 例子： csc /target:exe myProj.cs // 创建一个EXE文件 csc /target:winexe myProject.cs file://创建一个windows程序 csc /target:library myProject.cs file://创建一个代码库 csc /target:module myProject.cs file://创建一个模块
24	/resource	此选项和/linkresource正好相反。他的作用是把.Net资源文件嵌入到输出文件中，参数、用法都和/linkresource也相同，具体可参考前面/linkresource选项。
25	/unsafe	此选项是告诉编译器采用非安全模式编译文件 例子：用非安全模式编译my.cs csc /unsafe my.cs
26	/warn	使用本选项是在编译过程中采用什么等级的警告级别。警告级别含义：0关闭所有警告；1只显示严重警告；2级别为1的警告和某些不严重的警告；3级别为2的警告和某些不算非常严重的警告；4 级别为3的警告和信息警告

		例子：编译文件，不显示任何错误 csc /warn:0 my.cs
27	/warnaserror	告诉编译器把在编译中把所有的警告当成错误来处理。/warnaserror-是缺省选项，在该选项下编译中的警告不影响文件的输出。/warnaserror和/warnaserror+是一样的。 例子：编译文件，并在编译中把警告当成错误 csc /warnaserror myj.cs
28	/win32icon	在输出文件中插入一个图标文件（.ico）。从而在Windows中的资源管理器中就看到以此图标标识的文件了。 例子：csc /win32icon:myicon.ico my.cs
29	/win32res	在输出文件中添加一个win32的资源文件。此资源文件包括用户应用程序的版本信息或者位图（图标）信息。如果用户不指定/win32res，编译器将根据汇编版本生成版本信息。 例子：添加一个win32资源文件到输出文件中 csc /win32res:winrf.res mt.cs