gcc和g++是什么，有什么区别?

< 上一页 下一页 >

发展至今（2020 年 6 月份），GCC 编译器已经更新至 10.1.0 版本，其功能也由最初仅能编译 C 语言，扩增至可以编译多种编程语言，其中就包括 C++ 。

> 除此之外，当下的 GCC 编译器还支持编译 Go、Objective-C，Objective-C ++，Fortran，Ada，D 和 BRIG（HSAIL）等程序，甚至于 GCC 6 以及之前的版本还支持编译 Java 程序。但本教程主要讲解如何使用 GCC 编译器编译运行 C 和 C++ 程序，因此有关其它编程语言如何使用 GCC 编译器编译，将不再做具体讲解。

那么，在已编辑好 C 语言或者 C++ 代码的前提下，如何才能调用 GCC 编译器为我们编译程序呢？很简单，GCC 编译器已经为我们提供了调用它的接口，对于 C 语言或者 C++ 程序，可以通过执行 gcc 或者 g++ 指令来调用 GCC 编译器。

值得一提的是，实际使用中我们更习惯使用 gcc 指令编译 C 语言程序，用 g++ 指令编译 C++ 代码。需要强调的一点是，这并不是 gcc 和 g++ 的区别，gcc 指令也可以用来编译 C++ 程序，同样 g++ 指令也可以用于编译 C 语言程序。

那么，gcc 和 g++ 的区别是什么呢？接下来就给读者做详细的讲解。

实际上，只要是 GCC 支持编译的程序代码，都可以使用 gcc 命令完成编译。可以这样理解，gcc 是 GCC 编译器的通用编译指令，因为根据程序文件的后缀名，gcc 指令可以自行判断出当前程序所用编程语言的类别，比如：

- xxx.c：默认以编译 C 语言程序的方式编译此文件；
- xxx.cpp：默认以编译 C++ 程序的方式编译此文件。
- xxx.m：默认以编译 Objective-C 程序的方式编译此文件；
- xxx.go：默认以编译 Go 语言程序的方式编译此文件；

> 当然，gcc 指令也为用户提供了“手动指定代表编译方式”的接口，即使用 -x 选项。例如，gcc -xc xxx 表示以编译 C 语言代码的方式编译 xxx 文件；而 gcc -xc++ xxx 则表示以编译 C++ 代码的方式编译 xxx 文件。有关 -x 选项的用法，后续会给出具体样例。

但如果使用 g++ 指令，则无论目标文件的后缀名是什么，该指令都一律按照编译 C++ 代码的方式编译该文件。也就是说，对于 .c 文件来说，gcc 指令以 C 语言代码对待，而 g++ 指令会以 C++ 代码对待。但对于 .cpp 文件来说，gcc 和 g++ 都会以 C++ 代码的方式编译。

有读者可能会认为，C++ 兼容 C 语言，因此对于 C 语言程序来说，使用 gcc 编译还是使用 g++ 编译，应该没有什么区别，事实并非如此。严格来说，C++ 标准和 C 语言标准的语法要求是有区别的。举个例子：

```
//位于 demo.c 文件中
#include <stdio.h>
int main()
{
    const char * a = "abc";
    printStr(a);
    return;
}
int printStr(const char* str)
{
    printf(str);
}
```

如上所示，这是一段不规范的 C 语言代码。如果我们使用 gcc 指令编译，如下所示：

```
[root@bogon ~]# gcc -xc demo.c  #或者直接运行 gcc demo.c
[root@bogon ~]#
```

可以看到，该指令的执行过程并没有发生任何错误。而同样的程序，如果我们使用 g++ 指令编译：

```
[root@bogon ~]# g++ demo.c
demo.c: In function ‘int main()’:
demo.c:5: error: ‘printStr’ was not declared in this scope
demo.c:6: error: return-statement with no value, in function returning ‘int’
[root@bogon ~]#
```

可以看到，GCC 编译器发现了 3 处错误。显然，C++ 标准对代码书写规范的要求更加严格。

除此之外对于编译执行 C++ 程序，使用 gcc 和 g++ 也是有区别的。要知道，很多 C++ 程序都会调用某些标准库中现有的函数或者类对象，而单纯的 gcc 命令是无法自动链接这些标准库文件的。举个例子：

```
//demo.cpp
#include <iostream>
#include <string>
using namespace std;

int main(){
    string str ="C语言中文网";
    cout << str << endl;
```

```
09.     return 0;
10. }
```

这是一段很简单的 C++ 程序，其通过 <string> 头文件提供的 string 字符串类定义了一个字符串对象，随后使用 cout 输出流对象将其输出。对于这段 C++ 代码，如果我们使用 g++ 指令编译，如下所示：

```
[root@bogon ~]# g++ demo.cpp
[root@bogon ~]#
```

可以看到，整个编译过程没有报任何错误。但如果使用 gcc 指令：

```
[root@bogon ~]# gcc demo.cpp
/tmp/cclOnwra.o: In function `main':
demo.cpp:(.text+0x13): undefined reference to `std::allocator<char>::allocator()'
#省略了诸多错误信息
```

读者可自行编译，就可以看到很多报错信息。其根本原因就在于，该程序中使用了标准库 <iostream> 和<string> 提供的类对象，而 gcc 默认是无法找到它们的。

如果想使用 gcc 指令来编译执行 C++ 程序，需要在使用 gcc 指令时，手动为其添加 -lstdc++ -shared-libgcc 选项，表示 gcc 在编译 C++ 程序时可以链接必要的 C++ 标准库。也就是说，我们可以这样编译 demo.cpp 文件：

```
[root@bogon ~]# gcc -xc++ demo.cpp -lstdc++ -shared-libgcc
[root@bogon ~]#
```

由此，demo.cpp 就被成功的编译了。

> 读者可以这样认为，g++ 指令就等同于 `gcc -xc++ -lstdc++ -shared-libgcc` 指令。显然后者书写是非常麻烦的，大多数人会更喜欢前者。

对于 gcc 和 g++ 指令，还有其它更多细节方面的区别，这里不再做更多的赘述。读完本节，读者只需要知道，对于 C 语言程序的编译，我们应该使用 gcc 指令，而编译 C++ 程序则推荐使用 g++ 指令，这就足够了。

那么，编译成功的 C 语言或者 C++ 程序，该如何运行查看其执行结果呢？我们将在后续章节会大家做详细的讲解。

< 上一页 下一页 >

所有教程