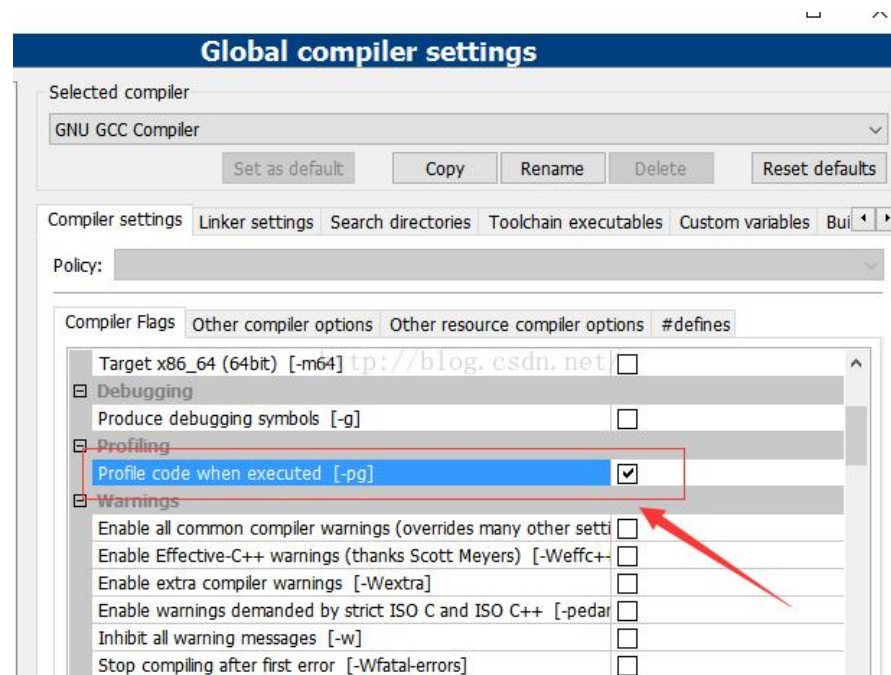


用 Code::Blocks Code profiler 插件剖析程序性能

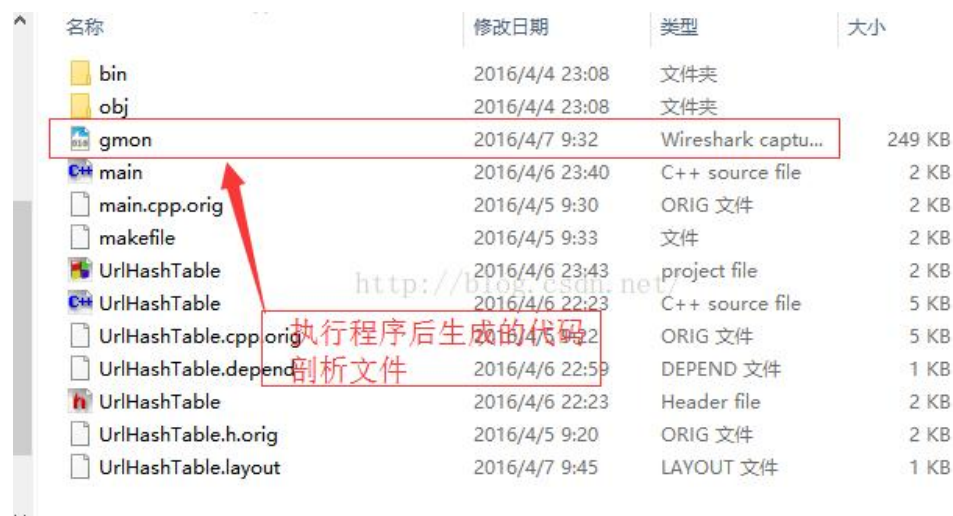
今天偶然发现 Code::Blocks 自带代码剖析插件:Code profiler, 之前一直命令行来使用, 现在发现 CB 有如此好的插件值得分享。其实这个插件也是调用 gprof 工具, 和 Linux 下的工具是一样的, 同样在 Linux 下命令行 gprof 也可以使用。下面以我的一个具体的程序剖析来说使用使用方法:

1.勾选编译器的-pg 选项:



2.编译工程, 然后运行一次程序:

注意:在添加-pg 选项编译后一定要运行一次, 要不然无法生成 gmon 文件, 从而导致无法剖析代码.



3.点击 Plugins-->Code profiler, 剖析结果如下:

C::B Profiler Results

Gprof's Output

Flat Profile Call Graph Misc

% time	cum. sec.	self sec.	calls	self ms/call	total ms/call	name
39.13	0.09	0.09				HashString(char const*)
30.43	0.16	0.07	2400000	0.00	0.00	UrlHash Table::AddUrl(UrlHash Table*, char const*)
17.39	0.20	0.04	2400004	0.00	0.00	main
8.70	0.22	0.02				std::string::operator+=(char)
4.35	0.23	0.01	1	10.00	10.00	UrlHash Table::DeleteHash Table(UrlHash Table*)
0.00	0.23	0.00	1	0.00	0.00	UrlHash Table::CreateHash(unsigned int)
0.00	0.23	0.00	1	0.00	0.00	UrlHash Table::DeleteUrl(UrlHash Table*, char const*)
0.00	0.23	0.00	1	0.00	0.00	UrlHash Table::SearchUrl(UrlHash Table*, char const*)
0.00	0.23	0.00	1	0.00	0.00	UrlHash Table::UrlHash Table()

% the percentage of the total running time of the program was used by this function

具体剖析结果就不一一介绍了，可以查看 gprof 的使用帮助。

转载于:<https://my.oschina.net/bufferoverflow/blog/689263>

文章知识点与官方知识档案匹配，可进一步学习相关知识

Ref: <https://blog.csdn.net/qianghaohao/article/details/51082930>