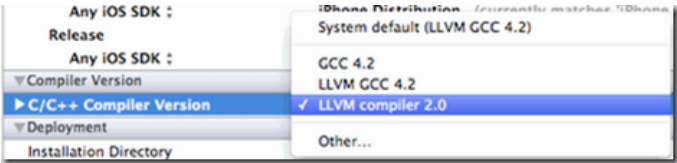


GCC，LLVM，Clang编译器对比

在XCode中，我们经常会看到这些编译选项（如下图），有些人可能会有些茫然，本文将对GCC4.2、LLVM GCC 4.2、LLVM compiler 2.0三个编译选项进行一个详细的介绍。



GCC

GCC（GNU Compiler Collection，GNU编译器套装），是一套由 GNU 开发的编程语言编译器。它是一套以 GPL 及 LGPL 许可证所发行的自由软件，也是 GNU计划的关键部分，亦是自由的类Unix及苹果电脑 Mac OS X 操作系统的标准编译器。

GCC 原名为 GNU C 语言编译器，因为它原本只能处理 C语言。GCC 很快地扩展，变得可处理 C++。之后也变得可处理 Fortran、Pascal、Objective-C、Java, 以及 Ada与其他语言。

LLVM

LLVM 是 Low Level Virtual Machine 的简称，这个库提供了与编译器相关的支持，能够进行程序语言的编译期优化、链接优化、在线编译优化、代码生成。简而言之，可以作为多种语言编译器的后台来使用。如果这样还比较抽象的话，介绍下 Clang 就知道了：Clang 是一个 C++ 编写、基于 LLVM、发布于 LLVM BSD 许可证下的 C/C++/Objective C/Objective C++ 编译器，其目标（之一）就是超越 GCC。

LLVM历史

Apple（包括中后期的NeXT）一直使用GCC作为官方的编译器。GCC作为开源世界的编译器标准一直做得不错，但Apple对编译工具会提出更高的要求。

一方面，是Apple对Objective-C语言（甚至后来对C语言）新增很多特性，但GCC开发者并不买Apple的帐——不给实现，因此索性后来两者分成两条分支分别开发，这也造成Apple的编译器版本远落后于GCC的官方版本。另一方面，GCC的代码耦合度太高，不好独立，而且越是后期的版本，代码质量越差，但Apple想做的很多功能（比如更好的IDE支持）需要模块化的方式来调用GCC，但GCC一直不给做。甚至最近，《GCC运行环境豁免条款（英文版）》从根本上限制了LLVM-GCC的开发。所以，这种不和让Apple一直在寻找一个高效的、模块化的、协议更放松的开源替代品，于是Apple请来了编译器高材生Chris Lattner（2000年，本科毕业的Chris Lattner像中国多数大学生一样，按部就班地考了GRE，最终前往UIUC（伊利诺伊大学厄巴纳香槟分校），开始了艰苦读计算机硕士和博士的生涯。在这阶段，他不仅周游美国各大景点，更是努力学习科学文化知识，翻译了“龙书”（《Compilers: Principles, Techniques, and Tools》），成了GPA牛人【注：最终学分积4.0满分】，以及不断地研究探索关于编译器的未知领域，发表了一篇又一篇的论文，是中国传统观念里的“三好学生”。他的硕士毕业论文提出了一套完整的在编译时、链接时、运行时甚至是在闲置时优化程序的编译思想，直接奠定了LLVM的基础。LLVM在他念博士时更加成熟，使用GCC作为前端来对用户程序进行语义分析产生IF（Intermediate Format），然后LLVM使用分析结果完成代码优化和生成。这项研究让他在2005年毕业时，成为小有名气的编译器专家，他也因此早早地被Apple相中，成为其编译器项目的骨干）。

刚进入Apple，Chris Lattner就大展身手：首先在OpenGL小组做代码优化，把LLVM运行时的编译架在OpenGL栈上，这样OpenGL栈能够产出更高效率的图形代码。如果显卡足够高级，这些代码会直接扔入GPU执行。但对于一些不支持全部OpenGL特性的显卡（比如当时的Intel GMA卡），LLVM则能够把这些指令优化成高效的CPU指令，使程序依然能够正常运行。这个强大的OpenGL实现被用在了后来发布的Mac OS X 10.5上。同时，LLVM的链接优化被直接加入到Apple的代码链接器上，而LLVM-GCC也被同步到使用GCC4代码。

Clang历史

Apple吸收Chris Lattner的目的要比改进GCC代码优化宏大得多——GCC系统庞大而笨重，而Apple大量使用的Objective-C在GCC中优先级很低。此外GCC作为一个纯粹的编译系统，与IDE配合得很差。加之许可证方面的要求，Apple无法使用LLVM 继续改进GCC的代码质量。于是，Apple决定从零开始写 C、C++、Objective-C语言的前端 Clang，完全替代掉GCC。

正像名字所写的那样，Clang只支持C，C++和Objective-C三种C家族语言。2007年开始开发，C编译器最早完成，而由于Objective-C相对简单，只是C语言的一个简单扩展，很多情况下甚至可以等价地改写为C语言对Objective-C运行库的函数调用，因此在2009年时，已经完全可以用于生产环境。C++的支持也热火朝天地进行着。

下面这张图将显示GCC、LLVM-GCC、LLVM Compiler这三个编译选项的不同点：

昵称: qoakzmxncb
园龄: 2年
粉丝: 2
关注: 0
+加关注

2013年4月						
<	一	二	三	四	五	六
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4
5	6	7	8	9	10	11

搜索

找找看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

更多链接

我的标签

arm64(1)

lame(1)

lame64位(1)

lame编译(1)

libmp3lame.a(1)

objc(1)

数字转中文大写(1)

随笔分类

开发之外(1)

随笔档案

2015年1月 (1)

2014年8月 (1)

2013年7月 (1)

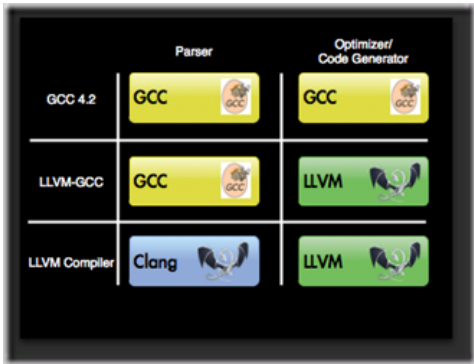
2013年4月 (1)

2013年1月 (1)

最新评论

1. Re:GCC，LLVM，Clang编译器对比

目前xcode4 以上是不是被替换成 LLVM-Clang 啦



对比

作为一种新的编译器，我们来看Clang和GCC各有什么优缺点：

Clang特性

- 1. 快：通过编译 OS X 上几乎包含了所有 C 头文件的 carbon.h 的测试，包括预处理 (Preprocess)，语法 (lex)，解析 (parse)，语义分析 (Semantic Analysis)，抽象语法树生成 (Abstract Syntax Tree) 的时间，Clang 是 Apple GCC 4.0 的 2.5x 快。(2007-7-25)
- 1. 内存占用小：Clang 内存占用是源码的 130%，Apple GCC 则超过 10x。
- 2. 诊断信息可读性强：我不会排版，推荐去网站观看。其中错误的语法不但有源码提示，还会在错误的调用和相关上下文的下方有~~~~~和^的提示，相比之下 GCC 的提示很天书。
- 3. GCC 兼容性。
- 4. 设计清晰简单，容易理解，易于扩展增强。与代码基础古老的 GCC 相比，学习曲线平缓。
- 5. 基于库的模块化设计，易于 IDE 集成及其他用途的重用。由于历史原因，GCC 是一个单一的可执行程序编译器，其内部完成了从预处理到最后代码生成的全部过程，中间诸多信息都无法被其他程序重用。Clang 将编译过程分成彼此分离的几个阶段，AST 信息可序列化。通过库的支持，程序能够获取到 AST 级别的信息，将大大增强对于代码的操控能力。对于 IDE 而言，代码补全、重构是重要的功能，然而如果没有底层的支持，只使用 tags 分析或是正则表达式匹配是很难达成的。

当然，GCC 也有其优势：

- 支持 JAVA/ADA/FORTRAN
- 当前的 Clang 的 C++ 支持落后于 GCC，参见http://clang.llvm.org/cxx_status.html。（近日 Clang 已经可以自编译，见http://www.phoronix.com/scan.php?page=news_item&px=Nzk2Mw）
- GCC 支持更多平台
- GCC 更流行，广泛使用，支持完备
- GCC 基于 C，不需要 C++ 编译器即可编译

要选择哪个

那么三个编译选项，要选择哪一个呢？目前不推荐使用老的GCC4.2，因为苹果不会维持它了，而且LLVM-GCC看起来会更好。在项目中途改编译选项可是一个大变动，所以，如果你要改，当然需要经过慎重完整的测试。

对新的项目而言，LLVM-GCC看起来应该是个安全的选择，苹果公司认为它够稳定够成熟，所以才把它当做Xcode 4的预设选项（你或许不会把稳定成熟这两个字眼跟Xcode 4本身画上等号），而且，既然选项使用的是GCC parser，向后兼容性应该没问题。

我说LLVM-GCC是个安全的选项，但我并不是指Clang/LLVM比较不安全，只是成熟度还没那么高效了，我将一些以前的代码拿到Xcode 4上，使用LLVM 2.0编译器重新编译，到目前为止还没发现任何问题。

参考文档：

<http://baike.baidu.com/view/4848.htm>
<http://hi.baidu.com/zhanghuikl/blog/item/71e8a6018172df0f728da53e.html>
<http://www.programmer.com.cn/9436/>

绿色通道：[好文要顶](#)[关注我](#)[收藏该文](#)[与我联系](#)

[qoakzmxncb](#)
[关注 - 0](#)
[粉丝 - 2](#)
[+加关注](#)

3

0

推荐

反对

(请您对文章做出评价)

« 上一篇：[如何判断自己是否到了该辞职的时候](#)
» 下一篇：[【转】My App Crashed, Now What? – Part 1](#)

2. Re:GCC, LLVM, Clang编译器对比

总结的不错，谢谢分享！

--hellmonky

阅读排行榜

1. GCC, LLVM, Clang编译器对比(5559)

2. 如何判断自己是否到了该辞职的时候(234)

3. 【转】iOS lame编译 arm64 armv7s armv7 x86_64 i386指令集(85)

4. 【转】My App Crashed, Now What? – Part 1(53)

5. 【OBJC】数字转中文大写(9)

评论排行榜

1. GCC, LLVM, Clang编译器对比(2)

推荐排行榜

1. GCC, LLVM, Clang编译器对比(3)