

LLVM IR 简介

魏恒峰

hfwei@nju.edu.cn

2022 年 12 月 19 日

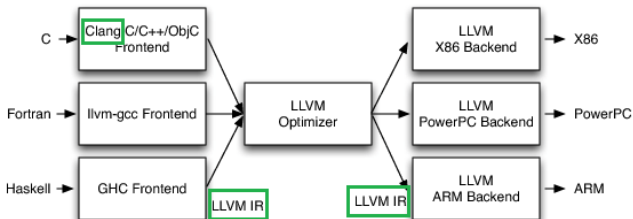


LLVM Overview

Latest LLVM Release!

2 / 22





https://clang.llvm.org



[Home](#)

[Info](#)

[Road](#)

[Tools](#)

[FAQs](#)

[Projects](#)

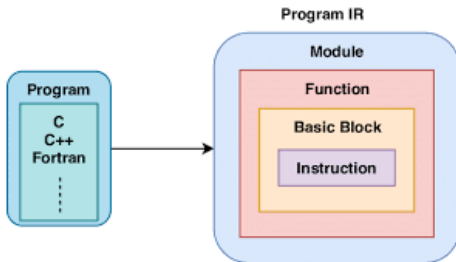
Clang: a C language family frontend for LLVM

The Clang project provides a language **front-end** and tooling infrastructure for languages in the C language family (C, C++, Objective C/C++, OpenCL, CUDA, and RenderScript) for the [LLVM](#) project. Both a GCC-compatible compiler driver (`clang`) and an MSVC-compatible compiler driver (`clang-cl.exe`) are provided. You can [get and build](#) the source today.

```
clang hello.c -o hello
```

hello @ CompilerExplorer

```
clang -Xclang -ast-dump -c hello.c
```



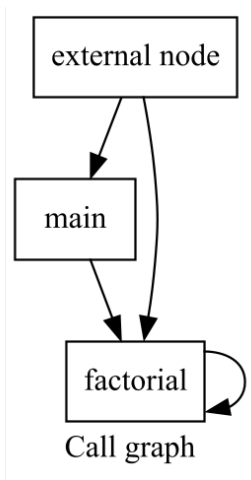
```

; Function Attrs: noinline nounwind optnone uwtable
define dso_local i32 @main(i32 %0, i8** %1) #0 {
    %3 = alloca i32, align 4
    %4 = alloca i32, align 4
    %5 = alloca i8**, align 8
    store i32 0, i32* %3, align 4
    store i32 %0, i32* %4, align 4
    store i8** %1, i8*** %5, align 8
    %6 = call i32 @factorial(i32 2)
    %7 = mul nsw i32 %6, 7
    %8 = icmp eq i32 %7, 42
    %9 = zext i1 %8 to i32
    ret i32 %9
}

```

`clang -S -emit-llvm factorial0.c f0-opt0.ll`

`clang -S -emit-llvm factorial0.c f0-opt1.ll -O1 -g0`



factorial1.c

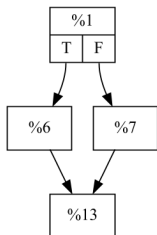
%2

CFG for 'main' function

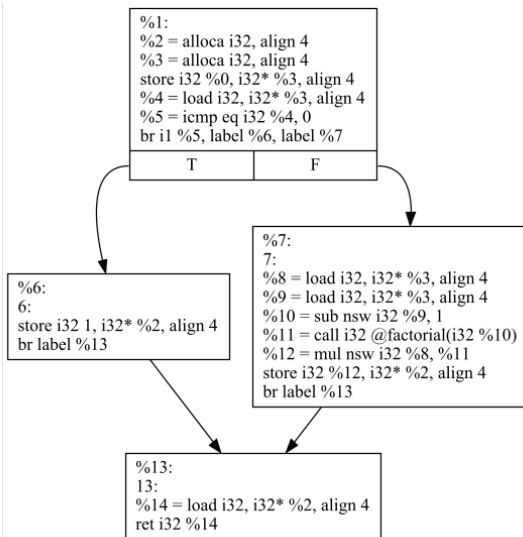
```
%2:  
%3 = alloca i32, align 4  
%4 = alloca i32, align 4  
%5 = alloca i8**, align 8  
store i32 0, i32* %3, align 4  
store i32 %0, i32* %4, align 4  
store i8** %1, i8*** %5, align 8  
%6 = call i32 @factorial(i32 2)  
%7 = mul nsw i32 %6, 7  
%8 = icmp eq i32 %7, 42  
%9 = zext i1 %8 to i32  
ret i32 %9
```

CFG for 'main' function

factorial1.c (opt0)



CFG for 'factorial' function



CFG for 'factorial' function

factorial1.c (opt0)

Instruction Reference

○ Terminator Instructions

- **'ret'** Instruction
- **'br'** Instruction
- **'switch'** Instruction
- **'indirectbr'** Instruction
- **'invoke'** Instruction
- **'callbr'** Instruction
- **'resume'** Instruction
- **'catchswitch'** Instruction
- **'catchret'** Instruction
- **'cleanupret'** Instruction
- **'unreachable'** Instruction

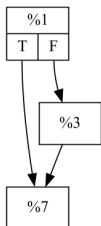
%2

CFG for 'main' function

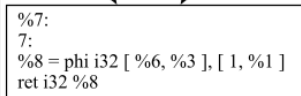
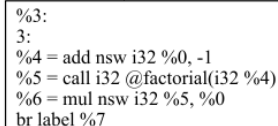
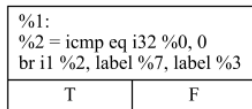
```
%2:  
%3 = call i32 @factorial(i32 2)  
%4 = mul nsw i32 %3, 7  
%5 = icmp eq i32 %4, 42  
%6 = zext i1 %5 to i32  
ret i32 %6
```

CFG for 'main' function

factorial1.c (opt1)



CFG for 'factorial' function



CFG for 'factorial' function

factorial1.c (opt1)

Single-Static Assignment Form and PHI

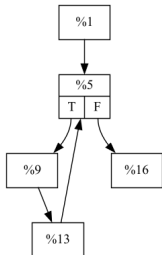
%2

CFG for 'main' function

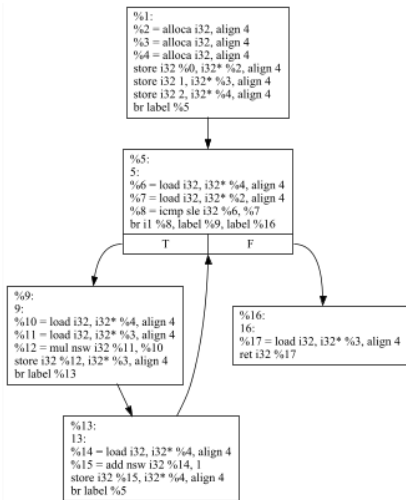
```
%2:  
%3 = alloca i32, align 4  
%4 = alloca i32, align 4  
%5 = alloca i8**, align 8  
store i32 0, i32* %3, align 4  
store i32 %0, i32* %4, align 4  
store i8** %1, i8*** %5, align 8  
%6 = call i32 @factorial(i32 2)  
%7 = mul nsw i32 %6, 7  
%8 = icmp eq i32 %7, 42  
%9 = zext i1 %8 to i32  
ret i32 %9
```

CFG for 'main' function

factorial2.c (opt0)



CFG for 'factorial' function



CFG for 'factorial' function

factorial2.c (opt0)

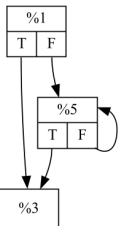
%2

CFG for 'main' function

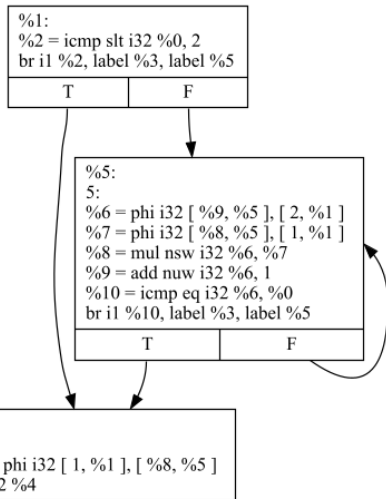
```
%2:  
%3 = call i32 @factorial(i32 2)  
%4 = mul nsw i32 %3, 7  
%5 = icmp eq i32 %4, 42  
%6 = zext i1 %5 to i32  
ret i32 %6
```

CFG for 'main' function

factorial2.c (opt1)




CFG for 'factorial' function



CFG for 'factorial' function

factorial2.c (opt1)

 <https://llvm.org/docs/LangRef.html>

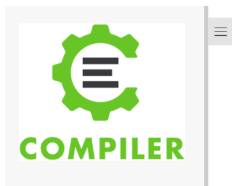


[LLVM Home](#) | [Documentation](#) » [Reference](#) »

LLVM Language Reference Manual

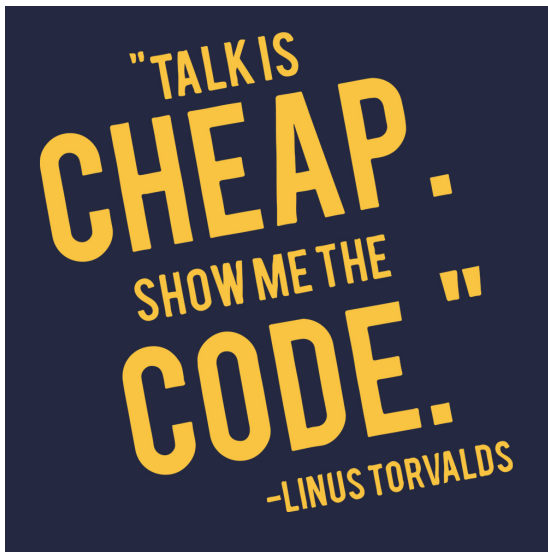
javacpp@github

JavaCPP Presets Platform For LLVM

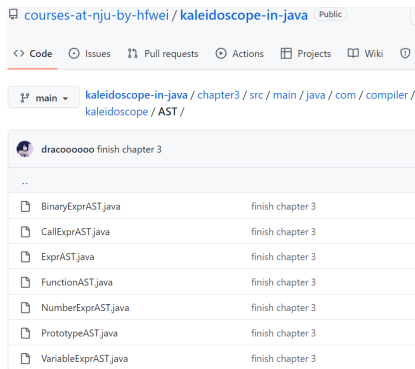


LLVM JAVA API使用手册

准备工作

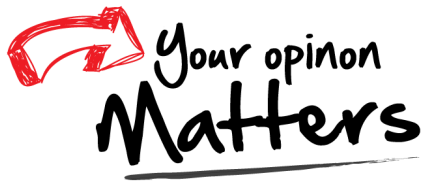


Kaleidoscope: Implementing a Language with LLVM



[kaleidoscope-in-java@github](#)

Thank
You!



Office 926

hfwei@nju.edu.cn