# How to use llvm-debuginfo-analyzer tool

Carlos Alberto Enciso



### How to use Ilvm-debuginfo-analyzer

- A. Introduction
  - 1. Common problems with debug information
  - 2. LLVM and debug information
- B. Ilvm-debuginfo-analyzer
  - 1. Introduction
  - 2. Print options
  - 3. Select options
  - 4. Compare options
- C. Future work



#### A. Introduction

- 1. Common problems with debug information
  - Does the debug information represent the original source
    - Which variables are dropped due to optimization
    - Why I cannot stop at a particular line
    - Which lines are associated to a specific code range
    - Size changes due to toolchain features
  - Semantic differences in the generated debug information
    - By different toolchain versions (same platform)
      - Clang 1.0.1 and 1.0.2
      - Clang and GCC
    - By same or different toolchain versions (different platforms)
      - Clang (Windows) and Clang (Linux)
      - Clang (Linux) and MSVC (Windows)



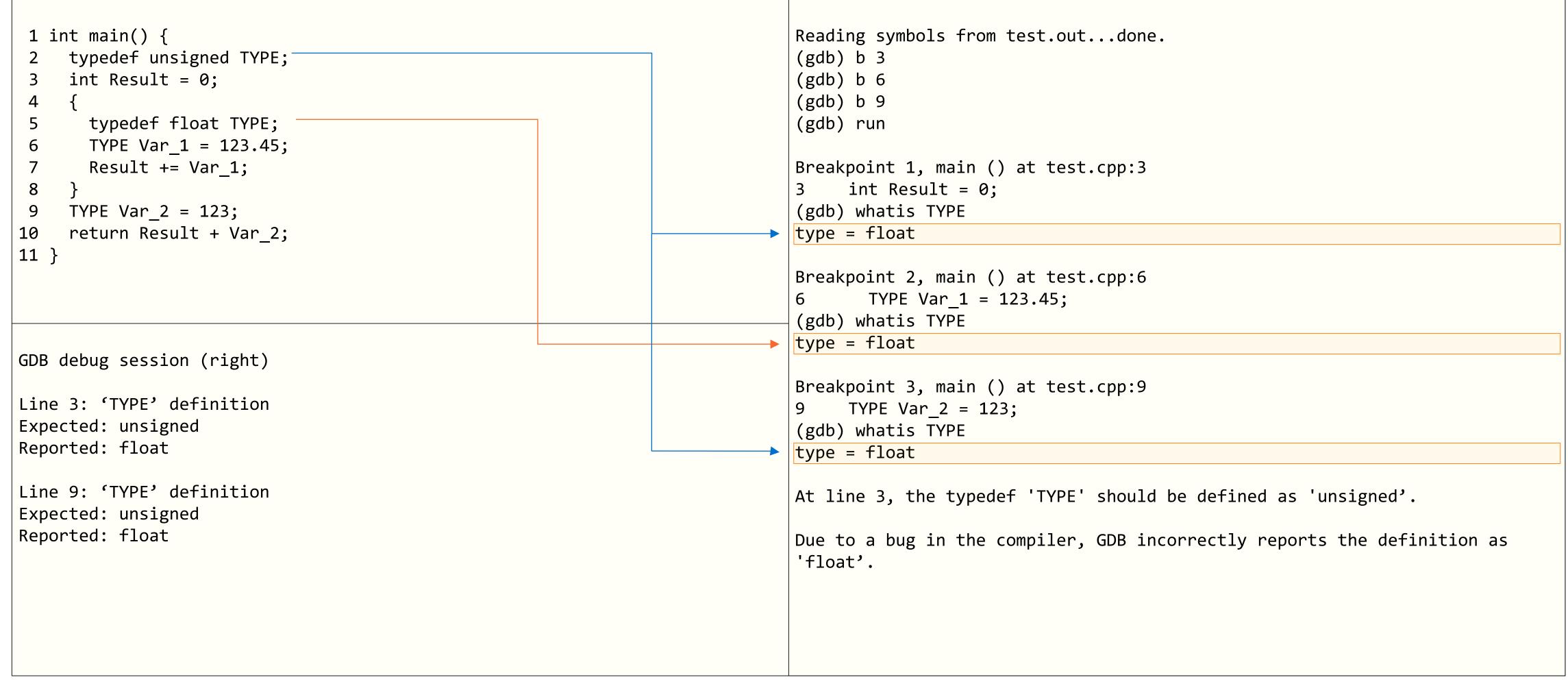
#### Debugger - Incorrect lexical scope for typedef 'TYPE' (PR44229)

```
Reading symbols from test.out...done.
1 int main() {
    typedef unsigned TYPE;
                                                                             (gdb) b 3
    int Result = 0;
                                                                             (gdb) b 6
                                                                             (gdb) b 9
      typedef float TYPE;
                                                                             (gdb) run
    TYPE Var_1 = 123.45;
       Result += Var_1;
                                                                             Breakpoint 1, main () at test.cpp:3
                                                                             3 int Result = 0;
                                                                             (gdb) whatis TYPE
    TYPE Var_2 = 123;
    return Result + Var_2;
                                                                             type = float
11 }
                                                                             Breakpoint 2, main () at test.cpp:6
                                                                                    TYPE Var_1 = 123.45;
                                                                             (gdb) whatis TYPE
                                                                             type = float
GDB debug session (right)
                                                                             Breakpoint 3, main () at test.cpp:9
Line 3: 'TYPE' definition
                                                                             9 TYPE Var_2 = 123;
Expected: unsigned
                                                                             (gdb) whatis TYPE
Reported: float
                                                                             type = float
Line 9: 'TYPE' definition
                                                                             At line 3, the typedef 'TYPE' should be defined as 'unsigned'.
Expected: unsigned
Reported: float
                                                                             Due to a bug in the compiler, GDB incorrectly reports the definition as
                                                                             'float'.
```

Example use case GDB debug session



### Debugger - Incorrect lexical scope for typedef 'TYPE' (PR44229)



Example use case GDB debug session



#### A. Introduction

#### 2. LLVM and debug information

- Several debug information formats
  - DWARF, CodeView
- Different binary file formats
  - ELF, COFF, PDB, Mach-O
- Different tools to print the debug information
  - Ilvm-dwarfdump, Ilvm-pdbutil, Ilvm-readelf
  - They use a close representation to the internal formats
  - Requires good knowledge of the format's specifications
- Understanding mappings between source code and debug information can be complex
- It is a problem commonly encountered when triaging LLVM's debug information issues

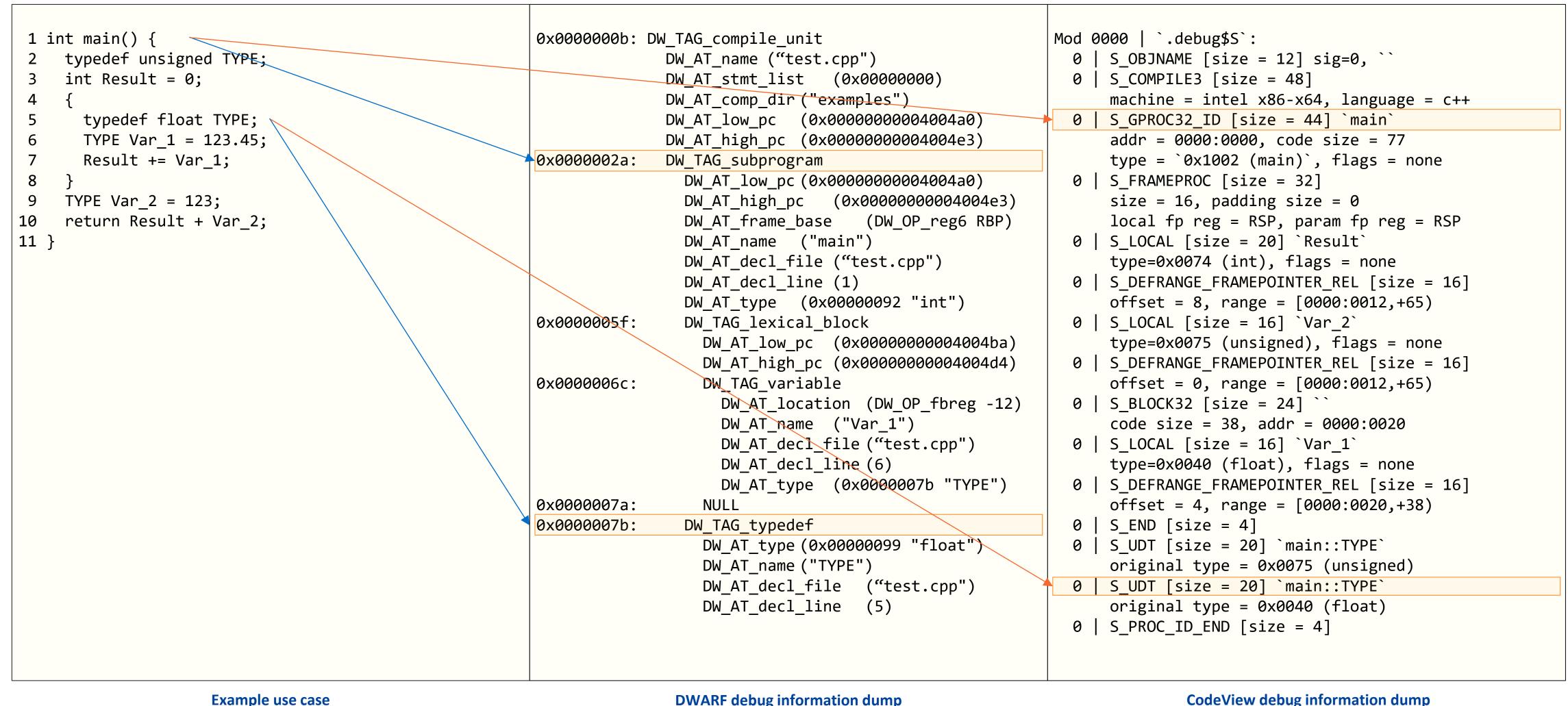
#### DWARF vs CodeView Debug Information

```
0x0000000b: DW TAG compile unit
                                                                                                                Mod 0000 | `.debug$S`:
 1 int main() {
     typedef unsigned TYPE;
                                                                      DW_AT_name ("test.cpp")
                                                                                                                  0 | S_OBJNAME [size = 12] sig=0, ``
                                                                                                                  0 | S COMPILE3 [size = 48]
     int Result = 0;
                                                                      DW_AT_stmt_list (0x0000000)
                                                                      DW AT comp dir ("examples")
                                                                                                                      machine = intel x86-x64, language = c++
                                                                                                                  0 | S_GPROC32_ID [size = 44] `main`
       typedef float TYPE;
                                                                      DW_AT_low_pc (0x00000000004004a0)
                                                                                                                      addr = 0000:0000, code size = 77
                                                                      DW_AT_high_pc (0x00000000004004e3)
      TYPE Var_1 = 123.45;
                                                                                                                      type = 0x1002 (main), flags = none
       Result += Var_1;
                                                                      DW_TAG_subprogram
                                                        0x0000002a:
                                                                                                                  0 | S_FRAMEPROC [size = 32]
                                                                        DW_AT_low_pc (0x00000000004004a0)
                                                                                                                      size = 16, padding size = 0
     TYPE Var_2 = 123;
                                                                        DW_AT_high_pc (0x00000000004004e3)
     return Result + Var_2;
                                                                                                                      local fp reg = RSP, param fp reg = RSP
                                                                        DW_AT_frame_base (DW_OP_reg6 RBP)
11 }
                                                                                                                  0 | S_LOCAL [size = 20] `Result`
                                                                        DW_AT_name ("main")
                                                                        DW_AT_decl_file ("test.cpp")
                                                                                                                      type=0x0074 (int), flags = none
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                                                                        DW_AT_decl_line (1)
                                                                        DW_AT_type (0x00000092 "int")
                                                                                                                      offset = 8, range = [0000:0012,+65)
                                                                        DW_TAG_lexical_block
                                                                                                                  0 | S_LOCAL [size = 16] `Var_2`
                                                        0x0000005f:
                                                                          DW AT low pc (0x00000000004004ba)
                                                                                                                      type=0x0075 (unsigned), flags = none
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                                                                          DW_AT_high_pc (0x00000000004004d4)
                                                                                                                      offset = 0, range = [0000:0012,+65)
                                                        0x0000006c:
                                                                          DW_TAG_variable
                                                                                                                  0 | S_BLOCK32 [size = 24] ``
                                                                            DW_AT_location (DW_OP_fbreg -12)
                                                                                                                      code size = 38, addr = 0000:0020
                                                                            DW_AT_name ("Var_1")
                                                                            DW_AT_decl_file ("test.cpp")
                                                                                                                  0 | S_LOCAL [size = 16] `Var_1`
                                                                                                                      type=0x0040 (float), flags = none
                                                                            DW_AT_decl_line (6)
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                                                                            DW_AT_type (0x0000007b "TYPE")
                                                                                                                      offset = 4, range = [0000:0020,+38)
                                                        0x0000007a:
                                                                          NULL
                                                                                                                  0 | S_END [size = 4]
                                                        0x0000007b:
                                                                        DW_TAG_typedef
                                                                          DW_AT_type (0x00000099 "float")
                                                                                                                  0 | S_UDT [size = 20] `main::TYPE`
                                                                          DW_AT_name ("TYPE")
                                                                                                                      original type = 0x0075 (unsigned)
                                                                          DW_AT_decl_file ("test.cpp")
                                                                                                                  0 | S_UDT [size = 20] `main::TYPE`
                                                                          DW_AT_decl_line
                                                                                                                      original type = 0x0040 (float)
                                                                                                                  0 | S_PROC_ID_END [size = 4]
```

Example use case DWARF debug information dump CodeView debug information dump



#### DWARF vs CodeView Debug Information



**CodeView debug information dump DWARF** debug information dump

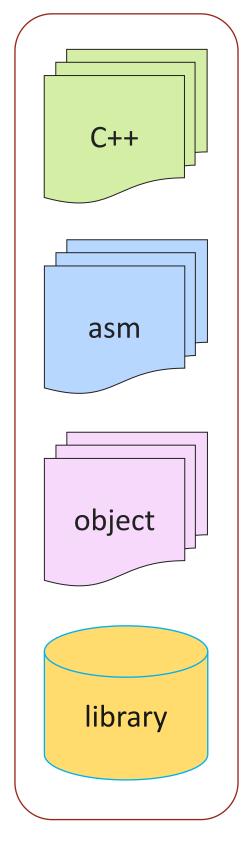


#### B. Ilvm-debuginfo-analyzer

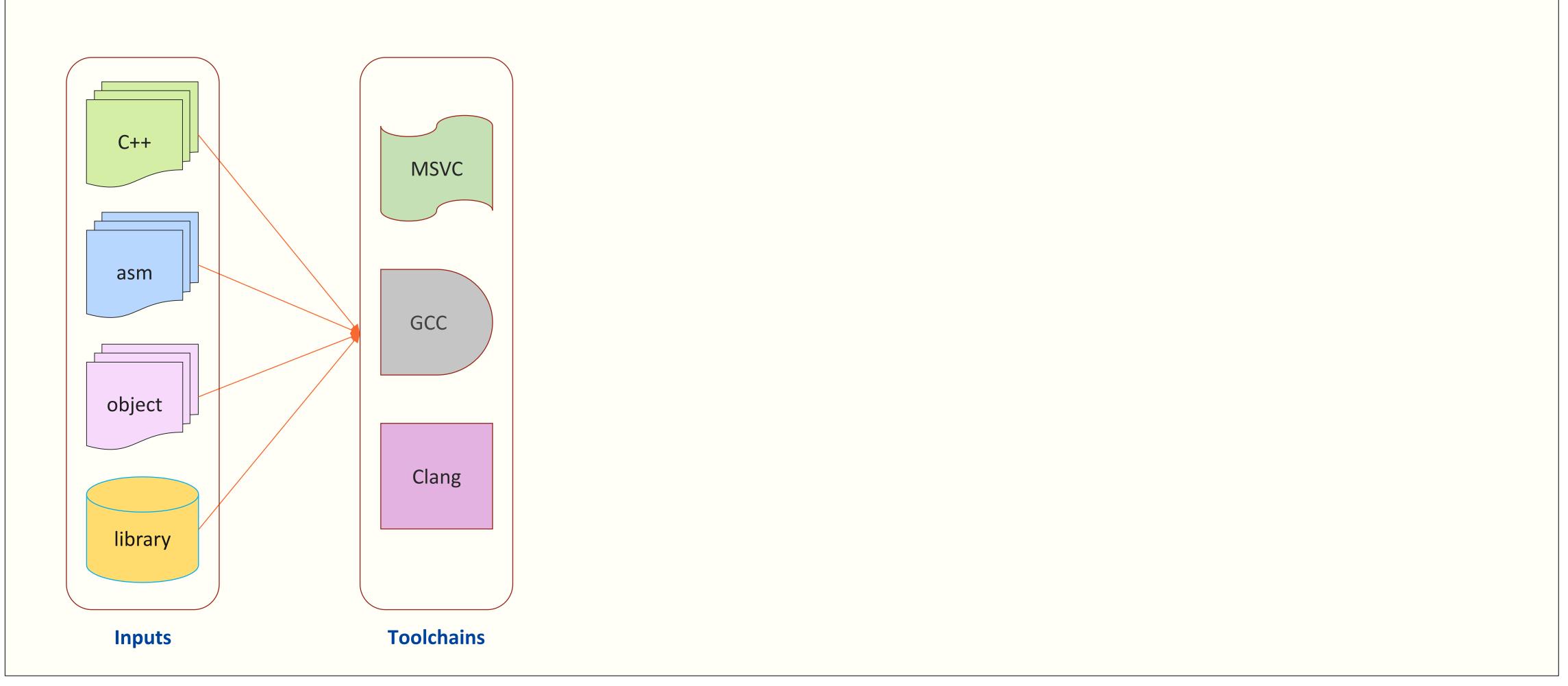
#### 1. Introduction

- Command line tool that processes the debugging information
  - Supported debug information formats: DWARF, CodeView
  - Supported binary file formats: ELF, COFF, PDB, Mach-O
- Produces a logical view, which is a high-level representation of the debug information
  - Composed of logical elements: scopes, types, symbols, and lines
  - Can display additional attributes: variable coverage factor, lexical block level, template argument encoding, etc.
- Key features
  - Uniform logical view regardless of the debug information and binary file formats
  - Logical lines associated to their logical scopes
  - Criteria used to select which logical elements to include in the logical view
  - Find semantic differences by comparing the logical views

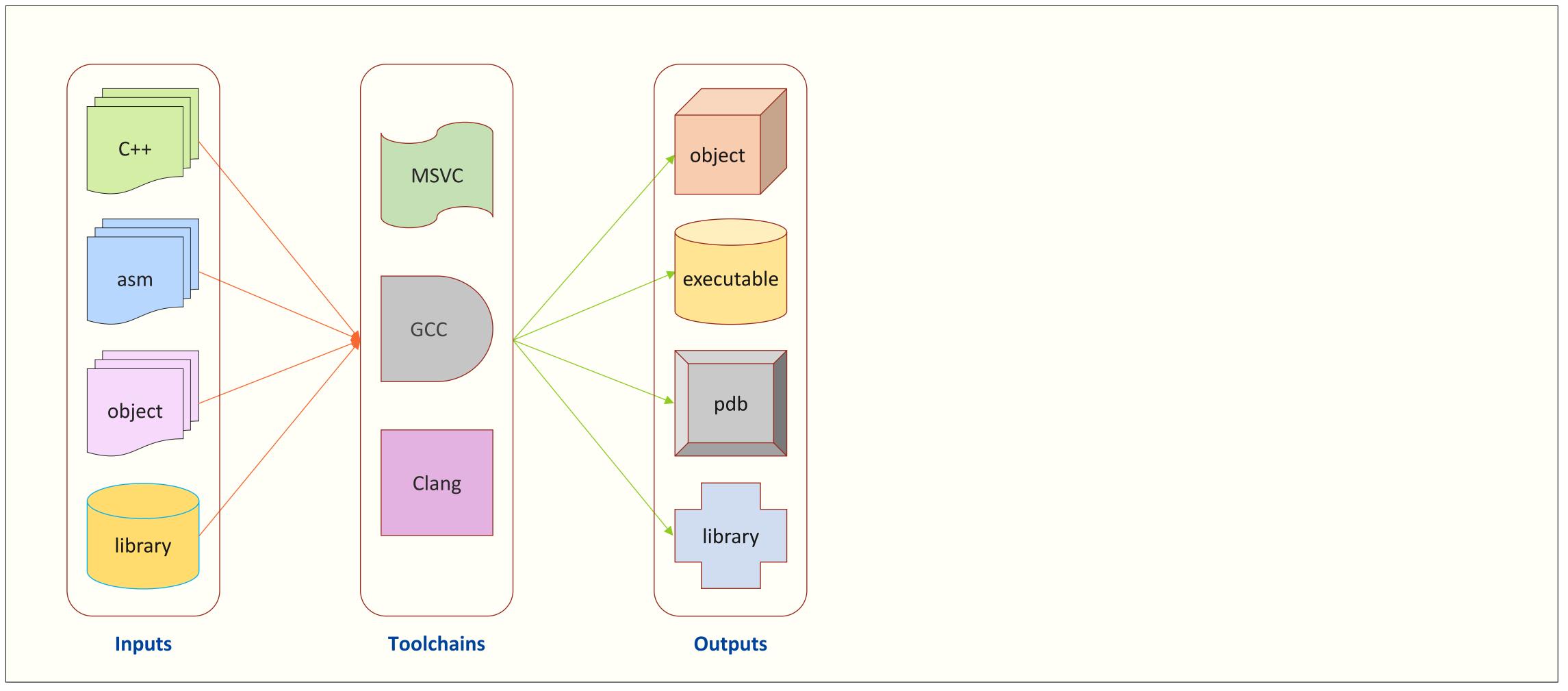




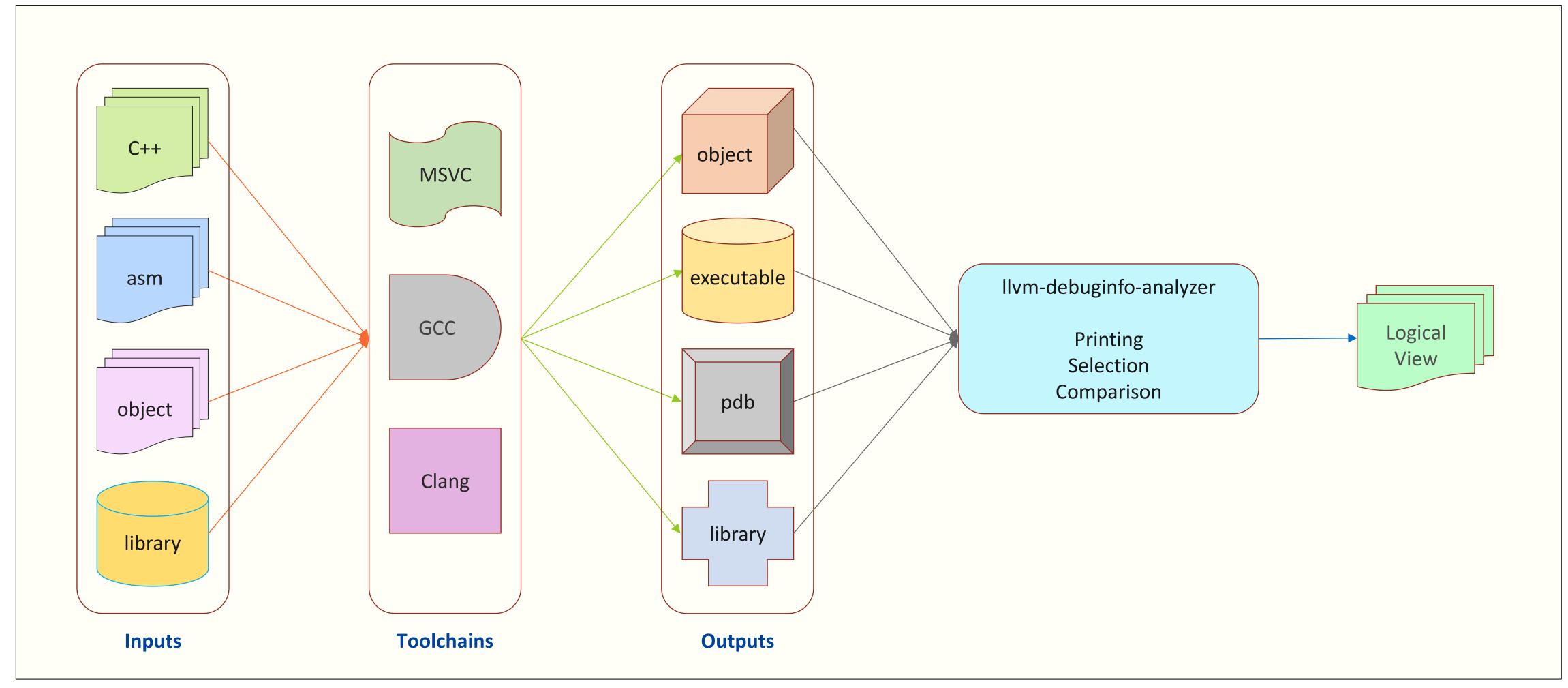
Inputs













#### Logical view

```
1 int main() {
                                                                               Logical View:
     typedef unsigned TYPE;
                                                                               [000]
                                                                                               {File} 'test.out'
                                                                                                {CompileUnit} 'test.cpp'
     int Result = 0;
                                                                               [001]
                                                                                                   {Function} extern 'main' -> 'int'
                                                                               [002]
                                                                                        1
 4
                                                                                                    {Block}
      typedef float TYPE;
                                                                               [003]
                                                                                                       {Variable} 'Var_1' -> 'TYPE'
      TYPE Var_1 = 123.45;
                                                                               [004]
       Result += Var_1;
                                                                               [004]
                                                                                                       {Line}
                                                                               [004]
                                                                                                       {Line}
     TYPE Var_2 = 123;
                                                                               [004]
                                                                                                       {Line}
     return Result + Var_2;
                                                                                                     {TypeAlias} 'TYPE' -> 'unsigned'
                                                                               [003]
11 }
                                                                                                     {Variable} 'Result' -> 'int'
                                                                               [003]
                                                                               [003]
                                                                                                    {TypeAlias} 'TYPE' -> 'float'
                                                                                                     {Variable} 'Var_2' -> 'TYPE'
                                                                               [003]
                                                                               [003]
                                                                                                     {Line}
                                                                               [003]
                                                                                                     {Line}
Type redefinition
                                                                               [003]
                                                                                                    {Line}
                                                                                                    {Line}
                                                                               [003]
                                                                                       10
Lexical Scope Level 3:
                                                                               [003]
                                                                                                     {Line}
                                                                                       10
TYPE -> unsigned, declared at line 2
                                                                               [002]
                                                                                       10
                                                                                                   {Line}
TYPE -> float, declared at line 5
                                                                               The logical scope at level 3, contains 2 different definitions for the
                                                                              typedef 'TYPE' with underlying type: 'unsigned' and 'float'.
```

Example use case Logical view



#### Logical view

```
1 int main() {
                                                                               Logical View:
     typedef unsigned TYPE;
                                                                               [000]
                                                                                               {File} 'test.out'
     int Result = 0;
                                                                                                {CompileUnit} 'test.cpp'
                                                                               [001]
                                                                                                   {Function} extern 'main' -> 'int'
                                                                               [002]
                                                                                        1
 4
                                                                                                     {Block}
      typedef float TYPE;
                                                                               [003]
                                                                                                       {Variable} 'Var_1' -> 'TYPE'
      TYPE Var_1 = 123.45;
                                                                               [004]
       Result += Var_1;
                                                                               [004]
                                                                                                       {Line}
                                                                               [004]
                                                                                                       {Line}
     TYPE Var_2 = 123;
                                                                               [004]
                                                                                                       {Line}
     return Result + Var_2;
                                                                                                     {TypeAlias} 'TYPE' -> 'unsigned'
                                                                               [003]
11 }
                                                                                                     {Variable} 'Result' -> 'int'
                                                                               [003]
                                                                                                     {TypeAlias} 'TYPE' -> 'float'
                                                                               [003]
                                                                                                     {Variable} 'Var_2' -> 'TYPE'
                                                                               [003]
                                                                                         9
                                                                               [003]
                                                                                                     {Line}
                                                                               [003]
                                                                                                     {Line}
Type redefinition
                                                                               [003]
                                                                                                     {Line}
                                                                               [003]
                                                                                                     {Line}
                                                                                        10
Lexical Scope Level 3:
                                                                               [003]
                                                                                                     {Line}
                                                                                        10
TYPE -> unsigned, declared at line 2
                                                                                                   {Line}
                                                                               [002]
                                                                                        10
TYPE -> float, declared at line 5
                                                                               The logical scope at level 3, contains 2 different definitions for the
                                                                               typedef 'TYPE' with underlying type: 'unsigned' and 'float'.
```

Example use case Logical view



#### DWARF vs Logical view vs CodeView

```
Mod 0000 | `.debug$S`:
0x0000000b: DW_TAG_compile_unit
                                                        Logical View:
              DW_AT_name ("test.cpp")
                                                        [000]
                                                                        {File} 'test.o'
                                                                                                                  0 | S OBJNAME [size = 12] sig=0, ``
              DW_AT_stmt_list (0x0000000)
                                                        [001]
                                                                          {CompileUnit} 'test.cpp'
                                                                                                                  0 | S_COMPILE3 [size = 48]
              DW_AT_comp_dir ("examples")
                                                        [002]
                                                                            {Function} extern 'main' -> 'int'
                                                                                                                      machine = intel x86-x64, language = c++
                                                                                                                  0 | S_GPROC32_ID [size = 44] `main`
              DW_AT_low_pc (0x00000000004004a0)
                                                        [003]
                                                                              {Block}
                                                                                {Variable} 'Var_1' -> 'TYPE'
                                                                                                                       addr = 0000:0000, code size = 77
              DW_AT_high_pc (0x00000000004004e3)
                                                        [004]
                                                        [004]
                                                                                                                      type = 0x1002 (main), flags = none
              DW_TAG_subprogram
0x0000002a:
                                                                                {Line}
                                                        [004]
                                                                                                                  0 | S_FRAMEPROC [size = 32]
                DW_AT_low_pc (0x00000000004004a0)
                                                                                {Line}
                                                                                                                      size = 16, padding size = 0
                                                        [004]
                                                                                {Line}
                DW_AT_high_pc (0x00000000004004e3)
                                                                                                                      local fp reg = RSP, param fp reg = RSP
                DW_AT_frame_base (DW_OP_reg6 RBP)
                                                        [003]
                                                                              {TypeAlias} 'TYPE' -> 'unsigned'
                                                                              {Variable} 'Result' -> 'int'
                                                                                                                  0 | S_LOCAL [size = 20] `Result`
                DW_AT_name ("main")
                                                        [003]
                DW_AT_decl_file ("test.cpp")
                                                                              {TypeAlias} 'TYPE' -> 'float'
                                                                                                                      type=0x0074 (int), flags = none
                                                        [003]
                                                                              {Variable} 'Var_2' -> 'TYPE'
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                DW_AT_decl_line (1)
                                                        [003]
                DW_AT_type (0x00000092 "int")
                                                        [003]
                                                                                                                      offset = 8, range = [0000:0012,+65)
                                                                              {Line}
                DW_TAG_lexical_block
                                                        [003]
                                                                                                                  0 | S_LOCAL [size = 16] `Var_2`
0x0000005f:
                                                                              {Line}
                                                                                                                      type=0x0075 (unsigned), flags = none
                  DW_AT_low_pc (0x00000000004004ba)
                                                        [003]
                                                                              {Line}
                                                                 10
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                  DW_AT_high_pc (0x00000000004004d4)
                                                        [003]
                                                                              {Line}
                                                                 10
                                                                                                                      offset = 0, range = [0000:0012,+65)
0x0000006c:
                  DW_TAG_variable
                                                        [003]
                                                                              {Line}
                                                                                                                  0 | S_BLOCK32 [size = 24] ``
                    DW_AT_location (DW_OP_fbreg -12)
                                                        [002]
                                                                 10
                                                                            {Line}
                                                                                                                       code size = 38, addr = 0000:0020
                    DW_AT_name ("Var_1")
                    DW_AT_decl_file ("test.cpp")
                                                                                                                  0 | S_LOCAL [size = 16] `Var_1`
                    DW_AT_decl_line (6)
                                                                                                                      type=0x0040 (float), flags = none
                                                                                                                  0 | S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
                    DW_AT_type (0x0000007b "TYPE")
                                                                                                                      offset = 4, range = [0000:0020,+38)
0x0000007a:
                  NULL
                                                                                                                  0 | S_END [size = 4]
                DW_TAG_typedef
0x0000007b:
                  DW_AT_type (0x00000099 "float")
                                                                                                                  0 | S_UDT [size = 20] `main::TYPE`
                  DW_AT_name ("TYPE")
                                                                                                                      original type = 0x0075 (unsigned)
                  DW_AT_decl_file ("test.cpp")
                                                                                                                  0 | S_UDT [size = 20] `main::TYPE`
                  DW AT decl line (5)
                                                                                                                      original type = 0x0040 (float)
                                                                                                                  0 | S_PROC_ID_END [size = 4]
```

DWARF debug information dump Logical View dump CodeView debug information dump



# DWARF vs Logical view vs CodeView

0x000000b: DW_TAG_compile_unit		Logical	Vi ew·		Mod 0000   `.debug\$S`:
DW_AT_name ("test.cpp")		[000]	L VICW.	{File} 'test.o'	0   S_OBJNAME [size = 12] sig=0, ``
	DW_AT_name ( cest.cpp ) DW_AT_stmt_list (0x00000000)	[001]		{CompileUnit} 'test.cpp'	0   S_COMPILE3 [size = 48]
	DW_AT_scmc_fist (0x0000000)  DW_AT_comp_dir("examples")	[002]	1	{Function} extern 'main' -> 'int'	machine = intel x86-x64, language = c++
	DW_AT_low_pc (0x0000000004004a0)	[003]		{Block}	0   S_GPROC32_ID [size = 44] `main`
	DW_AT_high_pc (0x000000000004004e3)	[004]	6	{Variable} 'Var_1' -> 'TYPE'	addr = 0000:0000, code size = 77
0x0000002a:	DW_TAG_subprogram	[004]	6	{Line}	type = `0x1002 (main)`, flags = none
onococca.	DW_AT_low_pc (0x0000000004004a0)	[004]	7	{Line}	0   S_FRAMEPROC [size = 32]
	DW_AT_high_pc (0x00000000000004004e3)	[004]	7	{Line}	size = 16, padding size = 0
	DW_AT_frame_base (DW_OP_reg6 RBP)	[003]	2	{TypeAlias} 'TYPE' -> 'unsigned'	local fp reg = RSP, param fp reg = RSP
	DW AT name ("main")	[003]	3	{Variable} 'Result' -> 'int'	0   S_LOCAL [size = 20] `Result`
	DW_AT_decl_file ("test.cpp")	[003]	5	{TypeAlias} 'TYPE' -> 'float'	type=0x0074 (int), flags = none
	DW_AT_decl_line (1)	[003]	9	{Variable} 'Var_2' -> 'TYPE'	0   S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
	DW_AT_type (0x00000092 "int")	[003]	1	{Line}	offset = 8, range = [0000:0012,+65)
0x0000005f:	DW_TAG_lexical_block	[003]	3	{Line}	0   S_LOCAL [size = 16] `Var_2`
	 DW_AT_low_pc (0x0000000004004ba)	[003]	9	{Line}	type=0x0075 (unsigned), flags = none
	DW_AT_high_pc (0x00000000004004d4)	[003]	10	{Line}	0   S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
0x0000006c:	DW_TAG_variable	[003]	10	{Line}	offset = 0, range = [0000:0012,+65)
	DW_AT_location (DW_OP_fbreg -12)	[002]	10	{Line}	0   S_BLOCK32 [size = 24] ``
	DW_AT_name ("Var_1")				code size = 38, addr = 0000:0020
	<pre>DW_AT_decl_file ("test.cpp")</pre>				0   S_LOCAL [size = 16] `Var_1`
	DW_AT_decl_line (6)				type=0x0040 (float), flags = none
	DW_AT_type (0x0000007b "TYPE")				0   S_DEFRANGE_FRAMEPOINTER_REL [size = 16]
0x0000007a:	NULL				offset = 4, range = [0000:0020,+38)
0x0000007b:	DW_TAG_typedef				0   S_END [size = 4]
	DW_AT_type (0x00000099 "float")				0   S_UDT [size = 20] `main::TYPE`
	DW_AT_name ("TYPE")				original type = 0x0075 (unsigned)
	<pre>DW_AT_decl_file ("test.cpp")</pre>				0   S_UDT [size = 20] `main::TYPE`
	DW_AT_decl_line (5)				original type = 0x0040 (float)
					0   S_PROC_ID_END [size = 4]

DWARF debug information dump Logical View dump CodeView debug information dump



# B. Ilvm-debuginfo-analyzer

#### 2. Print options

Print the logical views for:

e.g.

- DWARF 00 and DWARF 02
- CodeView O0 and CodeView O2
- DWARF O0 and CodeView O0
- DWARF O2 and CodeView O2



### --print (DWARF 00 and DWARF 02)

```
1 void bar(int Param);
3 void foo() {
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
7 }
Linux: clang test.cpp -c -g -00 -o dwarf-zero.o
Linux: clang test.cpp -c -g -02 -o dwarf-two.o
Print the logical views:
llvm-debuginfo-analyzer
  --attribute=format
  --print=scopes,symbols,types,lines,instructions
  dwarf-zero.o dwarf-two.o
or
                                                            4
llvm-debuginfo-analyzer
  --attribute=format
  --print=symbols,types,lines,instructions
  dwarf-zero.o dwarf-two.o
llvm-debuginfo-analyzer
  --attribute=format
                                                            7
  --print=elements
  dwarf-zero.o dwarf-two.o
```

```
Logical View:
                                                          Logical View:
   {File} 'dwarf-zero.o' -> elf64-x86-64
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
     {CompileUnit} 'test.cpp'
                                                                {CompileUnit} 'test.cpp'
       {Function} extern not_inlined 'foo' -> 'void'
                                                                 {Function} extern not_inlined 'bar' -> 'void'
                                                                    {Parameter} '' -> 'int'
         {Block}
           {Variable} 'Index' -> 'int'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
           {Line}
                                                                      {CallSiteParameter} '' -> 'void'
           {Code} 'movl $0x0, -0x4(%rbp)'
            {Line}
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
           {Code} 'cmpl $0x2, -0x4(%rbp)'
                                                                    {Block}
            {Line}
                                                                      {Variable} 'Index' -> 'int'
           {Code} 'jge 0x16'
           {Line}
                                                                      {Line}
           {Code} 'movl -0x4(%rbp), %edi'
                                                                      {Code} 'xorl %edi, %edi'
                                                                      {Code} 'callq 0x0'
           {Line}
           {Code} 'callq 0x0'
                                                                      {Code} 'movl $0x1, %edi'
                                                                      {Line}
            {Line}
                                                           5
           {Code} 'movl -0x4(%rbp), %eax'
                                                                      {Code} 'popq %rax'
            {Code} 'addl $0x1, %eax'
                                                                      {Code} 'jmp 0x0'
           {Code} 'movl %eax, -0x4(%rbp)'
                                                                      {Line}
            {Line}
                                                                    {Line}
           \{Code\} 'jmp -0x20'
                                                                    {Code} 'pushq %rax'
           {Line}
           {Code} 'addq $0x10, %rsp'
         {Line}
          {Code} 'pushq %rbp'
          {Code} 'movq
                         %rsp, %rbp'
                         $0x10, %rsp'
          {Code} 'subq
          {Code} 'popq
                         %rbp'
         {Code} 'retq'
         {Line}
```

Print: DWARF 00 Print: DWARF 02



### --print (DWARF 00 and DWARF 02)

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                                                                                         {File} 'dwarf-two.o' -> elf64-x86-64
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                                                                           {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                             {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                               {Parameter} '' -> 'int'
      bar(Index);
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                               {CallSite} 'bar' -> 'void'
7 }
                                                                       {Line}
                                                                       {Code} 'movl $0x0, -0x4(%rbp)'
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                                                                               {CallSite} 'bar' -> 'void'
Linux: clang test.cpp -c -g -00 -o dwarf-zero.o
                                                                       {Line}
Linux: clang test.cpp -c -g -02 -o dwarf-two.o
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                       {Code} 'cmpl $0x2, -0x4(%rbp)'
                                                                                                                               {Block}
                                                                       {Line}
                                                                       {Code} 'jge 0x16'
                                                                                                                                 {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                                                                                 {Line}
Print the logical views:
                                                                       {Code} 'movl -0x4(%rbp), %edi'
                                                                                                                                 {Code} 'xorl %edi, %edi'
                                                                                                                                 {Code} 'callq 0x0'
                                                                       {Line}
llvm-debuginfo-analyzer
                                                                       {Code} 'callq 0x0'
                                                                                                                                 {Code} 'movl $0x1, %edi'
  --attribute=format
                                                                       {Line}
                                                                                                                                 {Line}
  --print=scopes,symbols,types,lines,instructions
                                                                       {Code} 'movl -0x4(%rbp), %eax'
                                                                                                                                 {Code} 'popq %rax'
  dwarf-zero.o dwarf-two.o
                                                                       {Code} 'addl $0x1, %eax'
                                                                                                                                 {Code} 'jmp 0x0'
                                                                       {Code} 'movl %eax, -0x4(%rbp)'
                                                                                                                                 {Line}
                                                                       {Line}
                                                                                                                               {Line}
                                                                       {Code} 'jmp
                                                                                   -0x20'
                                                                                                                               {Code} 'pushq %rax'
llvm-debuginfo-analyzer
                                                                       {Line}
  --attribute=format
                                                                       {Code} 'addq $0x10, %rsp'
  --print=symbols,types,lines,instructions
                                                                     {Line}
  dwarf-zero.o dwarf-two.o
                                                                     {Code} 'pushq %rbp'
                                                                     {Code} 'movq
                                                                                    %rsp, %rbp'
                                                                     {Code} 'subq
                                                                                    $0x10, %rsp'
                                                                     {Code} 'popq
                                                                                    %rbp'
llvm-debuginfo-analyzer
                                                                     {Code} 'retq'
  --attribute=format
                                                                     {Line}
  --print=elements
  dwarf-zero.o dwarf-two.o
```

Print: DWARF 00 Print: DWARF 02



### --print (CodeView O0 and CodeView O2)

```
1 void bar(int Param);
3 void foo() {
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
7 }
Windows: clang test.cpp -c -g -00 -o codeview-zero.o
Windows: clang test.cpp -c -g -O2 -o codeview-two.o
Print the logical views:
llvm-debuginfo-analyzer
  --attribute=format
  --print=scopes,symbols,types,lines,instructions
  codeview-zero.o codeview-two.o
or
llvm-debuginfo-analyzer
  --attribute=format
  --print=symbols,types,lines,instructions
  codeview-zero.o codeview-two.o
llvm-debuginfo-analyzer
  --attribute=format
  --print=elements
  codeview-zero.o codeview-two.o
```

```
Logical View:
                                                         Logical View:
   {File} 'codeview-zero.o' -> COFF-x86-64
                                                             {File} 'codeview-two.o' -> COFF-x86-64
     {CompileUnit} 'test.cpp'
                                                               {CompileUnit} 'test.cpp'
       {Function} extern not inlined 'foo' -> 'void'
                                                                 {Function} extern not_inlined 'foo' -> 'void'
                                                                   {Block}
         {Block}
           {Variable} 'Index' -> 'int'
                                                                     {Line}
                                                          5
                                                                      {Code} 'xorl %ecx, %ecx'
           {Line}
           {Code} 'movl $0x0, 0x24(%rsp)'
                                                                     {Code} 'callq 0x0'
           {Code} 'cmpl $0x2, 0x24(%rsp)'
                                                                      {Code} 'movl $0x1, %ecx'
                                                                     {Code} 'addq $0x28, %rsp'
           {Code} 'jge 0x19'
                                                                      {Code} 'jmp 0x0'
5
           {Line}
           {Code} 'movl 0x24(%rsp), %ecx'
                                                                   {Line}
           {Code} 'callq 0x0'
                                                                   {Code} 'subq
                                                                                   $0x28, %rsp'
           {Line}
           {Code} 'movl 0x24(%rsp), %eax'
           {Code} 'addl $0x1, %eax'
           {Code} 'movl %eax, 0x24(%rsp)'
           {Code} 'jmp
                        -0x24'
         {Line}
         {Code} 'subq
                         $0x28, %rsp'
         {Line}
         {Code} 'addq
                         $0x28, %rsp'
         {Code} 'retq'
```

Print: CodeView O0 Print: CodeView O2



### --print (CodeView O0 and CodeView O2)

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                    Logical View:
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                        {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                          {CompileUnit} 'test.cpp'
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not inlined 'foo' -> 'void'
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                    {Block}
                                                                                                                              {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                                 {Line}
                                                                                                                     5
                                                                                                                                 {Code} 'xorl %ecx, %ecx'
7 }
                                                                      {Line}
                                                                      {Code} 'movl $0x0, 0x24(%rsp)'
                                                                                                                                 {Code} 'callq 0x0'
Windows: clang test.cpp -c -g -00 -o codeview-zero.o
                                                                       {Code} 'cmpl $0x2, 0x24(%rsp)'
                                                                                                                                 {Code} 'movl $0x1, %ecx'
                                                                                                                                 {Code} 'addq $0x28, %rsp'
Windows: clang test.cpp -c -g -O2 -o codeview-two.o
                                                                      {Code} 'jge 0x19'
                                                                                                                                 {Code} 'jmp 0x0'
                                                           5
                                                                      {Line}
                                                                      {Code} 'movl 0x24(%rsp), %ecx'
                                                                                                                              {Line}
                                                                      {Code} 'callq 0x0'
                                                                                                                              {Code} 'subq
                                                                                                                                              $0x28, %rsp'
Print the logical views:
                                                                      {Line}
                                                                      {Code} 'movl 0x24(%rsp), %eax'
llvm-debuginfo-analyzer
                                                                      {Code} 'addl $0x1, %eax'
  --attribute=format
                                                                       {Code} 'movl %eax, 0x24(%rsp)'
  --print=scopes,symbols,types,lines,instructions
                                                                      {Code} 'jmp
                                                                                   -0x24'
  codeview-zero.o codeview-two.o
                                                                    {Line}
                                                           3
                                                                    {Code} 'subq
                                                                                    $0x28, %rsp'
or
                                                                    {Line}
                                                                    {Code} 'addq
                                                                                    $0x28, %rsp'
llvm-debuginfo-analyzer
                                                                     {Code} 'retq'
  --attribute=format
  --print=symbols,types,lines,instructions
  codeview-zero.o codeview-two.o
llvm-debuginfo-analyzer
  --attribute=format
  --print=elements
  codeview-zero.o codeview-two.o
```

Print: CodeView O0 Print: CodeView O2



### --print (DWARF 00 and CodeView 00)

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                         {File} 'codeview-zero.o' -> COFF-x86-64
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                           {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                     {Block}
                                                                                                                               {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                  {Variable} 'Index' -> 'int'
7 }
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Code} 'movl $0x0, -0x4(%rbp)'
                                                                                                                                  {Code} 'movl $0x0, 0x24(%rsp)'
                                                                       {Line}
                                                                                                                                  {Code} 'cmpl $0x2, 0x24(%rsp)'
                                                                                                                                  {Code} 'jge 0x19'
                                                                       {Code} 'cmpl $0x2, -0x4(%rbp)'
                                                                       {Line}
                                                                                                                                  {Line}
                                                                                                                      5
                                                                       {Code} 'jge 0x16'
                                                                                                                                  {Code} 'movl 0x24(%rsp), %ecx'
                                                                       {Line}
                                                                                                                                  {Code} 'callq 0x0'
Print the logical views:
                                                                       {Code} 'movl -0x4(%rbp), %edi'
                                                                                                                                  {Line}
                                                                                                                                  {Code} 'movl 0x24(%rsp), %eax'
                                                                       {Line}
llvm-debuginfo-analyzer
                                                                       {Code} 'callq 0x0'
                                                                                                                                  {Code} 'addl $0x1, %eax'
  --attribute=format
                                                                                                                                  {Code} 'movl %eax, 0x24(%rsp)'
                                                                       {Line}
  --print=scopes,symbols,types,lines,instructions
                                                                       {Code} 'movl -0x4(%rbp), %eax'
                                                                                                                                  {Code} 'jmp
                                                                                                                                               -0x24'
  dwarf-zero.o codeview-zero.o
                                                                       {Code} 'addl $0x1, %eax'
                                                                                                                                {Line}
                                                                       {Code} 'movl %eax, -0x4(%rbp)'
                                                                                                                               {Code} 'subq
                                                                                                                                               $0x28, %rsp'
or
                                                                                                                               {Line}
                                                                       {Line}
                                                            4
                                                                                                                               {Code} 'addq
                                                                       \{Code\} 'jmp -0x20'
                                                                                                                                               $0x28, %rsp'
llvm-debuginfo-analyzer
                                                                       {Line}
                                                                                                                               {Code} 'retq'
  --attribute=format
                                                                       {Code} 'addq $0x10, %rsp'
  --print=symbols,types,lines,instructions
                                                                     {Line}
  dwarf-zero.o codeview-zero.o
                                                                     {Code} 'pushq %rbp'
                                                                                    %rsp, %rbp'
                                                                     {Code} 'movq
                                                                                     $0x10, %rsp'
                                                                     {Code} 'subq
                                                                     {Code} 'popq
                                                                                     %rbp'
llvm-debuginfo-analyzer
                                                                     {Code} 'retq'
  --attribute=format
                                                                     {Line}
                                                            7
  --print=elements
  dwarf-zero.o codeview-zero.o
```

Print: DWARF 00 Print: CodeView 00



### --print (DWARF 00 and CodeView 00)

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                         {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                           {CompileUnit} 'test.cpp'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                                                                                {Block}
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                  {Variable} 'Index' -> 'int'
7 }
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Code} 'movl $0x0, -0x4(%rbp)'
                                                                                                                                  {Code} 'movl $0x0, 0x24(%rsp)'
                                                                       {Line}
                                                                                                                                  {Code} 'cmpl $0x2, 0x24(%rsp)'
                                                                                                                                  {Code} 'jge 0x19'
                                                                       {Code} 'cmpl $0x2, -0x4(%rbp)'
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                      5
                                                                       {Code} 'jge 0x16'
                                                                                                                                  {Code} 'movl 0x24(%rsp), %ecx'
                                                                                                                                  {Code} 'callq 0x0'
                                                                       {Line}
Print the logical views:
                                                                       {Code} 'movl -0x4(%rbp), %edi'
                                                                                                                                  {Line}
                                                                                                                                  {Code} 'movl 0x24(%rsp), %eax'
                                                                       {Line}
llvm-debuginfo-analyzer
                                                                       {Code} 'callq 0x0'
                                                                                                                                  {Code} 'addl $0x1, %eax'
  --attribute=format
                                                                       {Line}
                                                                                                                                  {Code} 'movl %eax, 0x24(%rsp)'
  --print=scopes,symbols,types,lines,instructions
                                                                       {Code} 'movl -0x4(%rbp), %eax'
                                                                                                                                  {Code} 'jmp
                                                                                                                                               -0x24'
  dwarf-zero.o codeview-zero.o
                                                                       {Code} 'addl $0x1, %eax'
                                                                                                                                {Line}
                                                                       {Code} 'movl %eax, -0x4(%rbp)'
                                                                                                                               {Code} 'subq
                                                                                                                                               $0x28, %rsp'
or
                                                                       {Line}
                                                                                                                               {Line}
                                                            4
                                                                                                                               {Code} 'addq
                                                                       \{Code\} 'jmp -0x20'
                                                                                                                                               $0x28, %rsp'
llvm-debuginfo-analyzer
                                                                       {Line}
                                                                                                                               {Code} 'retq'
  --attribute=format
                                                                       {Code} 'addq $0x10, %rsp'
  --print=symbols,types,lines,instructions
                                                                     {Line}
  dwarf-zero.o codeview-zero.o
                                                                     {Code} 'pushq %rbp'
                                                                                    %rsp, %rbp'
                                                                     {Code} 'movq
                                                                                    $0x10, %rsp'
                                                                     {Code} 'subq
                                                                     {Code} 'popq
                                                                                     %rbp'
llvm-debuginfo-analyzer
                                                                     {Code} 'retq'
  --attribute=format
                                                                     {Line}
  --print=elements
  dwarf-zero.o codeview-zero.o
```

Print: DWARF 00 Print: CodeView 00



### --print (DWARF O2 and CodeView O2)

```
1 void bar(int Param);
3 void foo() {
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
7 }
Print the logical views:
llvm-debuginfo-analyzer
  --attribute=format
  --print=scopes,symbols,types,lines,instructions
  dwarf-two.o codeview-two.o
or
llvm-debuginfo-analyzer
  --attribute=format
  --print=symbols,types,lines,instructions
  dwarf-two.o codeview-two.o
llvm-debuginfo-analyzer
  --attribute=format
  --print=elements
  dwarf-two.o codeview-two.o
```

```
Logical View:
                                                         Logical View:
   {File} 'dwarf-two.o' -> elf64-x86-64
                                                             {File} 'codeview-two.o' -> COFF-x86-64
                                                               {CompileUnit} 'test.cpp'
     {CompileUnit} 'test.cpp'
       {Function} extern not_inlined 'bar' -> 'void'
                                                                 {Function} extern not_inlined 'foo' -> 'void'
         {Parameter} '' -> 'int'
                                                                   {Block}
       {Function} extern not_inlined 'foo' -> 'void'
                                                                     {Line}
         {CallSite} 'bar' -> 'void'
                                                                     {Code} 'xorl %ecx, %ecx'
           {CallSiteParameter} '' -> 'void'
                                                                     {Code} 'callq 0x0'
         {CallSite} 'bar' -> 'void'
                                                                     {Code} 'movl $0x1, %ecx'
           {CallSiteParameter} '' -> 'void'
                                                                     {Code} 'addq $0x28, %rsp'
         {Block}
                                                                     {Code} 'jmp 0x0'
           {Variable} 'Index' -> 'int'
                                                                   {Line}
                                                                   {Code} 'subq
           {Line}
                                                                                   $0x28, %rsp'
           {Code} 'xorl %edi, %edi'
           {Code} 'callq 0x0'
           {Code} 'movl $0x1, %edi'
           {Line}
5
           {Code} 'popq %rax'
           {Code} 'jmp 0x0'
           {Line}
         {Line}
         {Code} 'pushq %rax'
```

Print: DWARF O2 Print: CodeView O2



### --print (DWARF O2 and CodeView O2)

```
1 void bar(int Param);
                                                          Logical View:
                                                                                                                    Logical View:
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                        {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                          {CompileUnit} 'test.cpp'
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                    {Parameter} '' -> 'int'
      bar(Index);
                                                                                                                              {Block}
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                {Line}
                                                                    {CallSite} 'bar' -> 'void'
                                                                                                                                 {Code} 'xorl %ecx, %ecx'
7 }
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                                 {Code} 'callq 0x0'
                                                                    {CallSite} 'bar' -> 'void'
                                                                                                                                 {Code} 'movl $0x1, %ecx'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                                 {Code} 'addq $0x28, %rsp'
                                                                                                                                 {Code} 'jmp 0x0'
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                              {Line}
                                                                      {Line}
                                                                                                                              {Code} 'subq
                                                                                                                                              $0x28, %rsp'
Print the logical views:
                                                                      {Code} 'xorl %edi, %edi'
                                                                      {Code} 'callq 0x0'
llvm-debuginfo-analyzer
                                                                       {Code} 'movl $0x1, %edi'
  --attribute=format
                                                                      {Line}
  --print=scopes,symbols,types,lines,instructions
                                                                      {Code} 'popq %rax'
  dwarf-two.o codeview-two.o
                                                                       {Code} 'jmp 0x0'
                                                                      {Line}
or
                                                                    {Line}
                                                                    {Code} 'pushq %rax'
llvm-debuginfo-analyzer
  --attribute=format
  --print=symbols,types,lines,instructions
  dwarf-two.o codeview-two.o
llvm-debuginfo-analyzer
  --attribute=format
  --print=elements
  dwarf-two.o codeview-two.o
```

Print: DWARF O2 Print: CodeView O2



### B. Ilvm-debuginfo-analyzer

#### 3. Select options

Print selected logical elements using **single criteria** (list and view layout) for: e.g.

- DWARF 00 and DWARF 02
- CodeView O0 and CodeView O2

Print selected logical elements using **combined criteria** (list and view layout) for: e.g.

- DWARF 00 and DWARF 02
- CodeView O0 and CodeView O2

#### --select (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                         {File} 'dwarf-zero.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                         {CompileUnit} 'test.cpp'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                      4 {Line}
      bar(Index);
                                                                     {Block}
                                                                                                                         {Line}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                         {Line}
7 }
                                                                                                                         {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                        {Line}
                                                                                                                      4 {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Selection criteria:
                                                                     {Line}
  Index or 4
                                                           Logical View:
                                                                                                                     Logical View:
llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                         {File} 'dwarf-two.o'
  --select=Index --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                         {CompileUnit} 'test.cpp'
  --report=list
                                                                                                                      4 {Variable} 'Index' -> 'int'
                                                                   {Function} extern not_inlined 'bar' -> 'void'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
  dwarf-zero.o dwarf-two.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Select: list layout



#### --select (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                                                                                     Logical View:
                                                          Logical View:
                                                              {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                        {File} 'dwarf-zero.o'
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                                                                        {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                     4 {Line}
                                                                                                                        {Line}
      bar(Index);
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'}
                                                                                                                        {Line}
                                                                                                                        {Line}
7 }
                                                                      {Line}
                                                                      {Line}
                                                                                                                      4 {Line}
                                                                                                                      4 {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
Select logical elements:
                                                                      {Line}
                                                                    {Line}
Selection criteria:
                                                                    {Line}
  Index or 4
                                                          Logical View:
                                                                                                                     Logical View:
llvm-debuginfo-analyzer
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                        {File} 'dwarf-two.o'
  --select=Index --select=4
                                                                {CompileUnit} 'test.cpp'
                                                                                                                         {CompileUnit} 'test.cpp'
  --report=list
                                                                  {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                     4 {Variable} 'Index' -> 'int'
  --print=symbols,lines
                                                                    {Parameter} '' -> 'int'
  dwarf-zero.o dwarf-two.o
                                                                  {Function} extern not_inlined 'foo' -> 'void'/
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'}
                                                                      {Line}
                                                                       {Line}
                                                                      {Line}
                                                                    {Line}
                                                           3
```

(sn) systems

### --select (DWARF O0 and DWARF O2) - view layout

```
1 void bar(int Param);
                                                            Logical View:
                                                                                                                      Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-zero.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
                                                                   {Function} extern not inlined 'foo' -> 'void'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
       bar(Index);
                                                                      {Block}
                                                                                                                                {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                   {Variable} 'Index' -> 'int'
7 }
                                                                        {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                                                                                  {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
 Selection criteria:
                                                                     {Line}
  Index or 4
                                                                                                                      Logical View:
                                                            Logical View:
 llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-two.o'
  --select=Index --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
   --report=view
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
   --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
                                                                                                                                {Block}
  dwarf-zero.o dwarf-two.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                  {Variable} 'Index' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                      {CallSite} 'bar' -> 'void'
                                                                        {CallSiteParameter} '' -> 'void'
                                                                     {Block}
                                                                        {Variable} 'Index' -> 'int'
                                                                        {Line}
                                                                        {Line}
                                                                        {Line}
                                                                      {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Select: view layout



#### --select (DWARF O0 and DWARF O2) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                         {File} 'dwarf-zero.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                           {CompileUnit} 'test.cpp'
                                                                  {Function} extern not inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                     {Block}
                                                                                                                               {Block}
                                                                       {Variable} 'Index' -> 'int' }
                                                                                                                                  {Variable} 'Index' -> 'int'
7 }
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Selection criteria:
                                                                     {Line}
  Index or 4
                                                                                                                     Logical View:
                                                           Logical View:
llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                         {File} 'dwarf-two.o'
  --select=Index --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                           {CompileUnit} 'test.cpp'
  --report=view
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                  {Function} extern not_inlined 'bar' -> 'void'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
                                                                                                                               {Block}
  dwarf-zero.o dwarf-two.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                 {Variable} 'Index' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                            3
```

(sn) systems

### --select (CodeView O0 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                          Logical View:
                                                                                                                    Logical View:
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                        {File} 'codeview-zero.o'
                                                                                                                        {CompileUnit} 'test.cpp'
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                                                                        {Variable} 'Index' -> 'int'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                    {Block}
                                                                                                                     4 {Line}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                     4 {Line}
7 }
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
                                                                                                                    Logical View:
                                                          Logical View:
Select logical elements:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                        {File} 'codeview-two.o'
                                                                                                                        {CompileUnit} 'test.cpp'
                                                                {CompileUnit} 'test.cpp'
Selection criteria:
                                                                  {Function} extern not_inlined 'foo' -> 'void'
  Index or 4
                                                                    {Block}
                                                                      {Line}
11vm-debuginfo-analyzer
                                                                    {Line}
  --select=Index --select=4
  --report=list
  --print=symbols,lines
  codeview-zero.o codeview-two.o
```

CodeView O0 and CodeView O2 Print Select: list layout



### --select (CodeView O0 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                        Logical View:
                                                                                                               Logical View:
                                                           {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                   {File} 'codeview-zero.o'
                                                                                                                   {CompileUnit} 'test.cpp'
3 void foo() {
                                                             {CompileUnit} 'test.cpp'
                                                                                                                   {Variable} 'Index' -> 'int'
   for (int Index = 0; Index < 2; ++Index) {</pre>
                                                               {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                4 {Line}
      bar(Index);
                                                                 {Block}
                                                                   {Variable} 'Index' -> 'int'
                                                                                                                4 {Line}
7 }
                                                                   {Line} }
                                                                   {Line}
                                                                   {Line} }
                                                                 {Line}
                                                                 {Line}
                                                                                                               Logical View:
                                                        Logical View:
Select logical elements:
                                                                                                                   {File} 'codeview-two.o'
                                                           {File} 'codeview-two.o' -> COFF-x86-64
                                                             {CompileUnit} 'test.cpp'
                                                                                                                   {CompileUnit} 'test.cpp'
Selection criteria:
                                                               {Function} extern not_inlined 'foo' -> 'void'
                                                                  {Line}
 Index or 4
                                                                 {Block}
llvm-debuginfo-analyzer
                                                                 {Line}
  --select=Index --select=4
  --report=list
  --print=symbols,lines
  codeview-zero.o codeview-two.o
```

(sn) systems

#### --select (CodeView O0 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                                                                                     Logical View:
                                                          Logical View:
                                                                                                                         {File} 'codeview-zero.o'
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                                                                           {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                    {Block}
                                                                                                                               {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                                 {Variable} 'Index' -> 'int'
7 }
                                                                                                                                 {Line}
                                                                      {Line}
                                                                      {Line}
                                                                                                                                 {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
                                                          Logical View:
                                                                                                                     Logical View:
Select logical elements:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                         {File} 'codeview-two.o'
                                                                                                                           {CompileUnit} 'test.cpp'
                                                                {CompileUnit} 'test.cpp'
Selection criteria:
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
 Index or 4
                                                                    {Block}
                                                                                                                               {Block}
                                                                      {Line}
llvm-debuginfo-analyzer
                                                                    {Line}
  --select=Index --select=4
  --report=view
  --print=symbols,lines
  codeview-zero.o codeview-two.o
```

CodeView O0 and CodeView O2 Print Select: view layout



#### --select (CodeView O0 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                                                                               Logical View:
                                                        Logical View:
                                                           {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                   {File} 'codeview-zero.o'
3 void foo() {
                                                             {CompileUnit} 'test.cpp'
                                                                                                                     {CompileUnit} 'test.cpp'
                                                                                                                       {Function} extern not_inlined 'foo' -> 'void'
                                                               {Function} extern not_inlined 'foo' -> 'void'
   for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                 {Block}
                                                                                                                         {Block}
                                                                   {Variable} 'Index' -> 'int'}—
                                                                                                                           {Variable} 'Index' -> 'int'
7 }
                                                                   {Line} }—
                                                                                                                           {Line}
                                                                   {Line}
                                                                                                                           {Line}
                                                                   {Line} }
                                                                 {Line}
                                                                 {Line}
                                                        Logical View:
                                                                                                               Logical View:
Select logical elements:
                                                           {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                   {File} 'codeview-two.o'
                                                             {CompileUnit} 'test.cpp'
                                                                                                                     {CompileUnit} 'test.cpp'
Selection criteria:
                                                               {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                       {Function} extern not_inlined 'foo' -> 'void'
 Index or 4
                                                                 {Block}
                                                                                                                         {Block}
                                                                   {Line}
llvm-debuginfo-analyzer
                                                                 {Line}
  --select=Index --select=4
  --report=view
  --print=symbols,lines
  codeview-zero.o codeview-two.o
```

CodeView O0 and CodeView O2 Print Select: view layout



### --select (combined) (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                      Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-zero.o'
3 void foo() {
                                                                                                                          {CompileUnit} 'test.cpp'
                                                                 {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                       4 {Line}
      bar(Index);
                                                                                                                         {Line}
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                         {Line}
7 }
                                                                                                                         {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                       4 {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Combined selection criteria:
                                                                     {Line}
  LineDebug and 4
                                                                                                                      Logical View:
                                                           Logical View:
llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-two.o'
  --select-lines=LineDebug --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                          {CompileUnit} 'test.cpp'
  --report=list
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                          {Variable} 'Index' -> 'int'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
  dwarf-zero.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
Combined selection criteria:
                                                                       {CallSiteParameter} '' -> 'void'
  Variable and Index
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
llvm-debuginfo-analyzer
                                                                     {Block}
  --select-symbols=Variable --select=Index
                                                                       {Variable} 'Index' -> 'int'
  --report=list
                                                                       {Line}
  --print=symbols, lines
                                                                       {Line}
  dwarf-two.o
                                                                       {Line}
                                                                     {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Combined select: list layout



## --select (combined) (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                      Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-zero.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                          {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                       4 {Line}
                                                                                                                          {Line}
      bar(Index);
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                         {Line}
                                                                                                                         {Line}
7 }
                                                                       {Line}
                                                                       {Line}
                                                                                                                       4 {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Combined selection criteria:
                                                                     {Line}
  LineDebug and 4
                                                                                                                      Logical View:
                                                           Logical View:
llvm-debuginfo-analyzer
                                                                                                                          {File} 'dwarf-two.o'
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
  --select-lines=LineDebug --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                          {CompileUnit} 'test.cpp'
  --report=list
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                          {Variable} 'Index' -> 'int'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
  dwarf-zero.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'/
                                                                     {CallSite} 'bar' -> 'void'
Combined selection criteria:
                                                                       {CallSiteParameter} '' -> 'void'
  Variable and Index
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
llvm-debuginfo-analyzer
                                                                     {Block}
  --select-symbols=Variable --select=Index
                                                                       {Variable} 'Index' -> 'int'
  --report=list
                                                                       {Line}
  --print=symbols,lines
                                                                       {Line}
  dwarf-two.o
                                                                       {Line}
                                                                     {Line}
                                                            3
                                                                                                                                        Combined select: list layout
```



**Print** 

**DWARF 00 and DWARF 02** 

# --select (combined) (DWARF 00 and DWARF 02) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                      Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-zero.o'
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
3 void foo() {
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not inlined 'foo' -> 'void'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                                                                                {Block}
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                  {Line}
7 }
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Combined selection criteria:
                                                                     {Line}
  LineDebug and 4
                                                                                                                      Logical View:
                                                           Logical View:
llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-two.o'
  --select-lines=LineDebug --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
  --report=view
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
                                                                                                                                {Block}
  dwarf-zero.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                  {Variable} 'Index' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
Combined selection criteria:
                                                                       {CallSiteParameter} '' -> 'void'
  Variable and Index
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
llvm-debuginfo-analyzer
                                                                     {Block}
  --select-symbols=Variable --select=Index
                                                                       {Variable} 'Index' -> 'int'
  --report=view
                                                                       {Line}
  --print=symbols,lines
                                                                        {Line}
  dwarf-two.o
                                                                        {Line}
                                                                     {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Combined select: view layout



# --select (combined) (DWARF 00 and DWARF 02) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                      Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-zero.o'
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
3 void foo() {
                                                                                                                              {Function} extern not inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not inlined 'foo' -> 'void'
      bar(Index);
                                                                                                                                {Block}
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                  {Line}
7 }
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
Select logical elements:
                                                                       {Line}
                                                                     {Line}
Combined selection criteria:
                                                                     {Line}
  LineDebug and 4
                                                                                                                      Logical View:
                                                           Logical View:
llvm-debuginfo-analyzer
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                          {File} 'dwarf-two.o'
  --select-lines=LineDebug --select=4
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                            {CompileUnit} 'test.cpp'
  --report=view
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
  --print=symbols,lines
                                                                     {Parameter} '' -> 'int'
                                                                                                                                {Block}
  dwarf-zero.o
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                  {Variable} 'Index' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
Combined selection criteria:
                                                                       {CallSiteParameter} '' -> 'void'
  Variable and Index
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
llvm-debuginfo-analyzer
                                                                     {Block}
  --select-symbols=Variable --select=Index
                                                                       {Variable} 'Index' -> 'int'
  --report=view
                                                                       {Line}
  --print=symbols,lines
                                                                       {Line}
  dwarf-two.o
                                                                       {Line}
                                                                     {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Combined select: view layout



## --select (combined) (CodeView O0 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                                                                                         {File} 'codeview-zero.o'
                                                               {File} 'codeview-zero.o' -> COFF-x86-64
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                                                                         {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                      4 {Line}
                                                                                                                        {Line}
      bar(Index);
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
7 }
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                                     {Line}
                                                           Logical View:
                                                                                                                     Logical View:
Select logical elements:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                         {File} 'codeview-two.o'
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                         {CompileUnit} 'test.cpp'
Combined selection criteria:
                                                                   {Function} extern not_inlined 'foo' -> 'void'
  LineDebug and 4
                                                                     {Block}
                                                                      {Line}
llvm-debuginfo-analyzer
                                                                     {Line}
  --select-lines=LineDebug --select=4
  --report=list
  --print=symbols,lines
  codeview-zero.o
Combined selection criteria:
  Variable and Index
llvm-debuginfo-analyzer
  --select-symbols=Variable --select=Index
  --report=list
  --print=symbols,lines
  codeview-two.o
```

CodeView O0 and CodeView O2 Print Combined select: list layout



# --select (combined) (CodeView O0 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                         Logical View:
                                                                                                                 Logical View:
                                                            {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                    {File} 'codeview-zero.o'
                                                                                                                    {CompileUnit} 'test.cpp'
3 void foo() {
                                                              {CompileUnit} 'test.cpp'
                                                                {Function} extern not inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                 4 {Line}
                                                                                                                  4 {Line}
      bar(Index);
                                                                  {Block}
                                                                    {Variable} 'Index' -> 'int'
7 }
                                                                    {Line} }
                                                                    {Line}
                                                                    {Line}
                                                                  {Line}
                                                                  {Line}
                                                         Logical View:
                                                                                                                 Logical View:
Select logical elements:
                                                            {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                    {File} 'codeview-two.o'
                                                              {CompileUnit} 'test.cpp'
                                                                                                                    {CompileUnit} 'test.cpp'
Combined selection criteria:
                                                                {Function} extern not_inlined 'foo' -> 'void'
  LineDebug and 4
                                                                   {Line}
                                                                  {Block}
llvm-debuginfo-analyzer
                                                                  {Line}
  --select-lines=LineDebug --select=4
  --report=list
  --print=symbols,lines
  codeview-zero.o
Combined selection criteria:
  Variable and Index
llvm-debuginfo-analyzer
  --select-symbols=Variable --select=Index
  --report=list
  --print=symbols, lines
  codeview-two.o
```

(sn) systems

# --select (combined) (CodeView O0 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Logical View:
                                                               {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                         {File} 'codeview-zero.o'
                                                                                                                           {CompileUnit} 'test.cpp'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not inlined 'foo' -> 'void'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                     {Block}
                                                                                                                               {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                 {Variable} 'Index' -> 'int'
7 }
                                                                                                                                 {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                                 {Line}
                                                                       {Line}
                                                                     {Line}
                                                                     {Line}
                                                           Logical View:
                                                                                                                     Logical View:
Select logical elements:
                                                               {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                         {File} 'codeview-two.o'
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                           {CompileUnit} 'test.cpp'
Combined selection criteria:
                                                                   {Function} extern not_inlined 'foo' -> 'void'
  LineDebug and 4
                                                                     {Block}
                                                                       {Line}
llvm-debuginfo-analyzer
                                                                     {Line}
  --select-lines=LineDebug --select=4
  --report=view
  --print=symbols,lines
  codeview-zero.o
Combined selection criteria:
  Variable and Index
llvm-debuginfo-analyzer
  --select-symbols=Variable --select=Index
  --report=view
  --print=symbols,lines
  codeview-two.o
```

CodeView O0 and CodeView O2 Print Combined select: view layout



## --select (combined) (CodeView O0 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                                                                                 Logical View:
                                                         Logical View:
                                                            {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                     {File} 'codeview-zero.o'
                                                                                                                       {CompileUnit} 'test.cpp'
3 void foo() {
                                                              {CompileUnit} 'test.cpp'
                                                                {Function} extern not inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                         {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                  {Block}
                                                                                                                           {Block}
                                                                                                                             {Variable} 'Index' -> 'int'
                                                                    {Variable} 'Index' -> 'int'
7 }
                                                                    {Line} }
                                                                                                                             {Line}
                                                                    {Line}
                                                                                                                             {Line}
                                                                    {Line} }
                                                                  {Line}
                                                                  {Line}
                                                                                                                 Logical View:
                                                         Logical View:
Select logical elements:
                                                            {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                     {File} 'codeview-two.o'
                                                              {CompileUnit} 'test.cpp'
                                                                                                                       {CompileUnit} 'test.cpp'
Combined selection criteria:
                                                                {Function} extern not_inlined 'foo' -> 'void'
  LineDebug and 4
                                                                   {Line}
                                                                  {Block}
llvm-debuginfo-analyzer
                                                                  {Line}
  --select-lines=LineDebug --select=4
  --report=view
  --print=symbols,lines
  codeview-zero.o
Combined selection criteria:
  Variable and Index
llvm-debuginfo-analyzer
  --select-symbols=Variable --select=Index
  --report=view
  --print=symbols,lines
  codeview-two.o
```

CodeView O0 and CodeView O2 Print Combined select: view layout

# B. Ilvm-debuginfo-analyzer

#### 4. Compare options

Find semantic differences by comparing logical views (list and view layout) for: e.g.

- DWARF O0 and DWARF O2
- CodeView O0 and CodeView O2
- DWARF O0 and CodeView O0
- DWARF O2 and CodeView O2



# --compare (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                                                                                     Reference: 'dwarf-zero.o'
                                                           Logical View:
                                                              {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                     Target:
                                                                                                                                'dwarf-two.o'
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                                                                     (7) Missing Lines:
                                                                  {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                     {Block}
                                                                                                                            {Line}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                             {Line}
                                                                                                                             {Line}
7 }
                                                                       {Line}
                                                                       {Line}
                                                                                                                            {Line}
                                                                       {Line}
                                                                                                                            {Line}
                                                                                                                            {Line}
                                                                      {Line}
                                                                                                                         7 {Line}
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                                                                     (3) Added Scopes:
Compare the logical views:
                                                                                                                             {CallSite} 'bar' -> 'void'
                                                                      {Line}
                                                                                                                             {CallSite} 'bar' -> 'void'
                                                                    {Line}
llvm-debuginfo-analyzer
                                                                                                                     + 1 {Function} extern not_inlined 'bar' -> 'void'
                                                                    {Line}
  --compare=symbols,lines
  --report=list
                                                                                                                     (5) Added Symbols:
                                                           Logical View:
  --print=symbols,lines
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                             {CallSiteParameter} '' -> 'void'
  dwarf-zero.o dwarf-two.o
                                                                {CompileUnit} 'test.cpp'
                                                                                                                             {CallSiteParameter} '' -> 'void'
                                                                  {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                             {Parameter} '' -> 'int'
                                                                    {Parameter} '' -> 'int'
                                                                                                                             {Parameter} '' -> 'int'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                             {Parameter} '' -> 'int'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                    {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                           3
```

DWARF 00 and DWARF 02 Print Compare: list layout



# --compare (DWARF 00 and DWARF 02) - list layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Reference: 'dwarf-zero.o'
                                                              {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                                'dwarf-two.o'
                                                                                                                     Target:
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                     (7) Missing Lines:
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                     {Block}
                                                                                                                            {Line}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                            {Line}
                                                                                                                            {Line}
7 }
                                                                       {Line}
                                                                       {Line}
                                                                                                                            {Line}
                                                                       {Line}
                                                                                                                            {Line}
                                                                                                                        7 {Line}
                                                                      {Line}
                                                                                                                        7 {Line}
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                     (3) Added Scopes:
Compare the logical views:
                                                                       {Line}
                                                                                                                            {CallSite} 'bar' -> 'void'
                                                                                                                            {CallSite} 'bar' -> 'void'
                                                                     {Line}
llvm-debuginfo-analyzer
                                                                                                                       1 {Function} extern not_inlined 'bar' -> 'void'
                                                                     {Line}
  --compare=symbols,lines
  --report=list
                                                                                                                     (5) Added Symbols:
                                                           Logical View:
  --print=symbols,lines
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                             {CallSiteParameter} '' -> 'void'
  dwarf-zero.o dwarf-two.o
                                                                                                                             {CallSiteParameter} '' -> 'void'
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                             {Parameter} '' -> 'int'
                                                                    {Parameter} '' -> 'int'
                                                                                                                             {Parameter} '' -> 'int'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                             {Parameter} '' -> 'int'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                    {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                           3
```

DWARF 00 and DWARF 02 Print Compare: list layout



#### --compare (DWARF O0 and DWARF O2) - view layout

```
Reference: 'dwarf-zero.o'
1 void bar(int Param);
                                                            Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                      Target:
                                                                                                                                  'dwarf-two.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                      Logical View:
                                                                                                                          {File} 'dwarf-zero.o'
       bar(Index);
                                                                      {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                            {CompileUnit} 'test.cpp'
7 }
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
                                                                        {Line}
                                                                        {Line}
                                                                                                                                {Block}
                                                                                                                                   {Variable} 'Index' -> 'int'
                                                                        {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                       {Line}
                                                                                                                                   {Line}
                                                                       {Line}
                                                                                                                                   {Line}
Compare the logical views:
                                                                       {Line}
                                                                                                                                   {Line}
                                                                                                                                   {Line}
                                                                     {Line}
 llvm-debuginfo-analyzer
                                                                     {Line}
                                                                                                                                   {Line}
   --compare=symbols, lines
                                                                                                                                   {Line}
  --report=view
                                                                                                                                {CallSite} 'bar' -> 'void'
                                                            Logical View:
   --print=symbols,lines
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                                  {CallSiteParameter} '' -> 'void'
  dwarf-zero.o dwarf-two.o
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                                  {Parameter} '' -> 'int'
                                                                                                                                {CallSite} 'bar' -> 'void'
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                     {Parameter} '' -> 'int'
                                                                                                                                   {CallSiteParameter} '' -> 'void'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                  {Parameter} '' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
                                                                                                                                {Line}
                                                                       {CallSiteParameter} '' -> 'void'
                                                                                                                                {Line}
                                                                      {CallSite} 'bar' -> 'void'
                                                                                                                              {Function} extern not_inlined 'bar' -> 'void'
                                                                        {CallSiteParameter} '' -> 'void'
                                                                                                                                {Parameter} '' -> 'int'
                                                                     {Block}
                                                                        {Variable} 'Index' -> 'int'
                                                                        {Line}
                                                                        {Line}
                                                                        {Line}
                                                                      {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Compare: view layout



#### --compare (DWARF 00 and DWARF 02) - view layout

```
Reference: 'dwarf-zero.o'
1 void bar(int Param);
                                                           Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                      Target:
                                                                                                                                  'dwarf-two.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                      Logical View:
    for (int Index = 0; Index < 2; ++Index) {</pre>
      bar(Index);
                                                                                                                          {File} 'dwarf-zero.o'
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                            {CompileUnit} 'test.cpp'
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
7 }
                                                                       {Line}
                                                                       {Line}
                                                                                                                                {Block}
                                                                                                                                  {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                       {Line}
                                                                                                                                  {Line}
                                                                       {Line}
                                                                                                                                  {Line}
Compare the logical views:
                                                                       {Line}
                                                                                                                                  {Line}
                                                                     {Line}
                                                                                                                                  {Line}
llvm-debuginfo-analyzer
                                                                                                                                  {Line}
                                                                     {Line}
  --compare=symbols,lines
                                                                                                                                  {Line}
  --report=view
                                                                                                                                {CallSite} 'bar' -> 'void'
                                                           Logical View:
  --print=symbols,lines
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                                  {CallSiteParameter} '' -> 'void'
  dwarf-zero.o dwarf-two.o
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                                  {Parameter} '' -> 'int'
                                                                   {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                                {CallSite} 'bar' -> 'void'
                                                                     {Parameter} '' -> 'int'
                                                                                                                                  {CallSiteParameter} '' -> 'void'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                  {Parameter} '' -> 'int'
                                                                     {CallSite} 'bar' -> 'void'
                                                                                                                                {Line}
                                                                       {CallSiteParameter} '' -> 'void'
                                                                                                                                {Line}
                                                                     {CallSite} 'bar' -> 'void'
                                                                                                                              {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                                {Parameter} '' -> 'int'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                       {Line}
                                                                       {Line}
                                                                     {Line}
                                                            3
```

DWARF 00 and DWARF 02 Print Compare: view layout



# --compare (CodeView O0 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                          Logical View:
                                                                                                                    Reference: 'codeview-zero.o'
                                                                                                                               'codeview-two.o'
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                    Target:
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                    (3) Missing Lines:
      bar(Index);
                                                                    {Block}
                                                                                                                            {Line}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                            {Line}
                                                                                                                       7 {Line}
7 }
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
                                                          Logical View:
Compare the logical views:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                {CompileUnit} 'test.cpp'
llvm-debuginfo-analyzer
                                                                  {Function} extern not_inlined 'foo' -> 'void'
  --compare=symbols,lines
                                                                    {Block}
  --report=list
                                                                      {Line}
  --print=symbols,lines
                                                                    {Line}
  codeview-zero.o codeview-two.o
```

CodeView O0 and CodeView O2 Print Compare: list layout



# --compare (CodeView O0 and CodeView O2) - list layout

```
Reference: 'codeview-zero.o'
1 void bar(int Param);
                                                          Logical View:
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                                'codeview-two.o'
                                                                                                                     Target:
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                     (3) Missing Lines:
      bar(Index);
                                                                    {Block}
                                                                                                                            {Line}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                            {Line}
7 }
                                                                                                                            {Line}
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
                                                          Logical View:
Compare the logical views:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                {CompileUnit} 'test.cpp'
llvm-debuginfo-analyzer
                                                                  {Function} extern not_inlined 'foo' -> 'void'
  --compare=symbols,lines
                                                                    {Block}
  --report=list
                                                                      {Line}
  --print=symbols,lines
                                                                    {Line}
  codeview-zero.o codeview-two.o
```

CodeView O0 and CodeView O2 Print Compare: list layout



#### --compare (CodeView O0 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                                                                                     Reference: 'codeview-zero.o'
                                                           Logical View:
                                                               {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                                 'codeview-two.o'
                                                                                                                     Target:
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                   {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                     Logical View:
                                                                                                                         {File} 'codeview-zero.o'
      bar(Index);
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                           {CompileUnit} 'test.cpp'
7 }
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                       {Line}
                                                                       {Line}
                                                                                                                               {Block}
                                                                                                                                 {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                     {Line}
                                                                                                                                 {Line}
                                                                     {Line}
                                                                                                                                 {Line}
                                                                                                                                 {Line}
                                                           Logical View:
                                                                                                                               {Line}
Compare the logical views:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                               {Line}
                                                                 {CompileUnit} 'test.cpp'
llvm-debuginfo-analyzer
                                                                   {Function} extern not_inlined 'foo' -> 'void'
  --compare=symbols,lines
                                                                     {Block}
  --report=view
                                                                       {Line}
  --print=symbols,lines
                                                                     {Line}
  codeview-zero.o codeview-two.o
```

(sn) systems

# --compare (CodeView O0 and CodeView O2) - view layout

```
Reference: 'codeview-zero.o'
1 void bar(int Param);
                                                           Logical View:
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                                'codeview-two.o'
                                                                                                                     Target:
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                  {Function} extern not_inlined 'foo' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                     Logical View:
                                                                                                                         {File} 'codeview-zero.o'
      bar(Index);
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                           {CompileUnit} 'test.cpp'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
7 }
                                                                       {Line}
                                                                       {Line}
                                                                                                                               {Block}
                                                                                                                                 {Variable} 'Index' -> 'int'
                                                                       {Line}
                                                                     {Line}
                                                                                                                                 {Line}
                                                                     {Line}
                                                                                                                                 {Line}
                                                                                                                                 {Line}
                                                           Logical View:
                                                                                                                               {Line}
Compare the logical views:
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                               {Line}
                                                                {CompileUnit} 'test.cpp'
llvm-debuginfo-analyzer
                                                                  {Function} extern not_inlined 'foo' -> 'void'
  --compare=symbols,lines
                                                                     {Block}
  --report=view
                                                                      {Line}
  --print=symbols,lines
                                                                     {Line}
  codeview-zero.o codeview-two.o
```

#### --compare (DWARF 00 and CodeView 00) - list layout

```
1 void bar(int Param);
                                                                                                                    Reference: 'dwarf-zero.o'
                                                          Logical View:
                                                              {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                               'codeview-zero.o'
                                                                                                                    Target:
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                    (1) Missing Scopes:
                                                                                                                        3 {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
7 }
                                                                                                                    (1) Missing Symbols:
                                                                      {Line}
                                                                      {Line}
                                                                                                                        4 {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                                                                    (1) Missing Lines:
                                                                      {Line}
                                                                                                                    - 7 {Line}
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                                                                    (1) Added Scopes:
Compare the logical views:
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
                                                                      {Line}
                                                                    {Line}
llvm-debuginfo-analyzer
                                                                                                                    (1) Added Symbols:
                                                                    {Line}
  --compare=symbols,lines
                                                                                                                            {Variable} 'Index' -> 'int'
  --report=list
                                                          Logical View:
  --print=symbols,lines
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
  dwarf-zero.o codeview-zero.o
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
```

DWARF 00 and CodeView 00 Print Compare: list layout



#### --compare (DWARF 00 and CodeView 00) - list layout

```
1 void bar(int Param);
                                                                                                                    Reference: 'dwarf-zero.o'
                                                          Logical View:
                                                              {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                                'codeview-zero.o'
                                                                                                                    Target:
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                     (1) Missing Scopes:
                                                                                                                        3 {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                     (1) Missing Symbols:
7 }
                                                                      {Line}
                                                                      {Line}
                                                                                                                        4 {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                                                                     (1) Missing Lines:
                                                                      {Line}
                                                                                                                        7 {Line}
                                                                      {Line}
                                                                      {Line}
                                                                                                                     (1) Added Scopes:
                                                                      {Line}
Compare the logical views:
                                                                                                                            {Function} extern not inlined 'foo' -> 'void'
                                                                      {Line}
                                                                    {Line}
llvm-debuginfo-analyzer
                                                                                                                     (1) Added Symbols:
                                                                    {Line}
  --compare=symbols,lines
                                                                                                                            {Variable} 'Index' -> 'int'
  --report=list
                                                          Logical View:
  --print=symbols,lines
                                                              {File} 'codeview-zero.o' -> COFF-x86-64
  dwarf-zero.o codeview-zero.o
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                      {Line}
                                                                      {Line}
                                                                    {Line}
                                                                    {Line}
```

DWARF 00 and CodeView 00 Print Compare: list layout

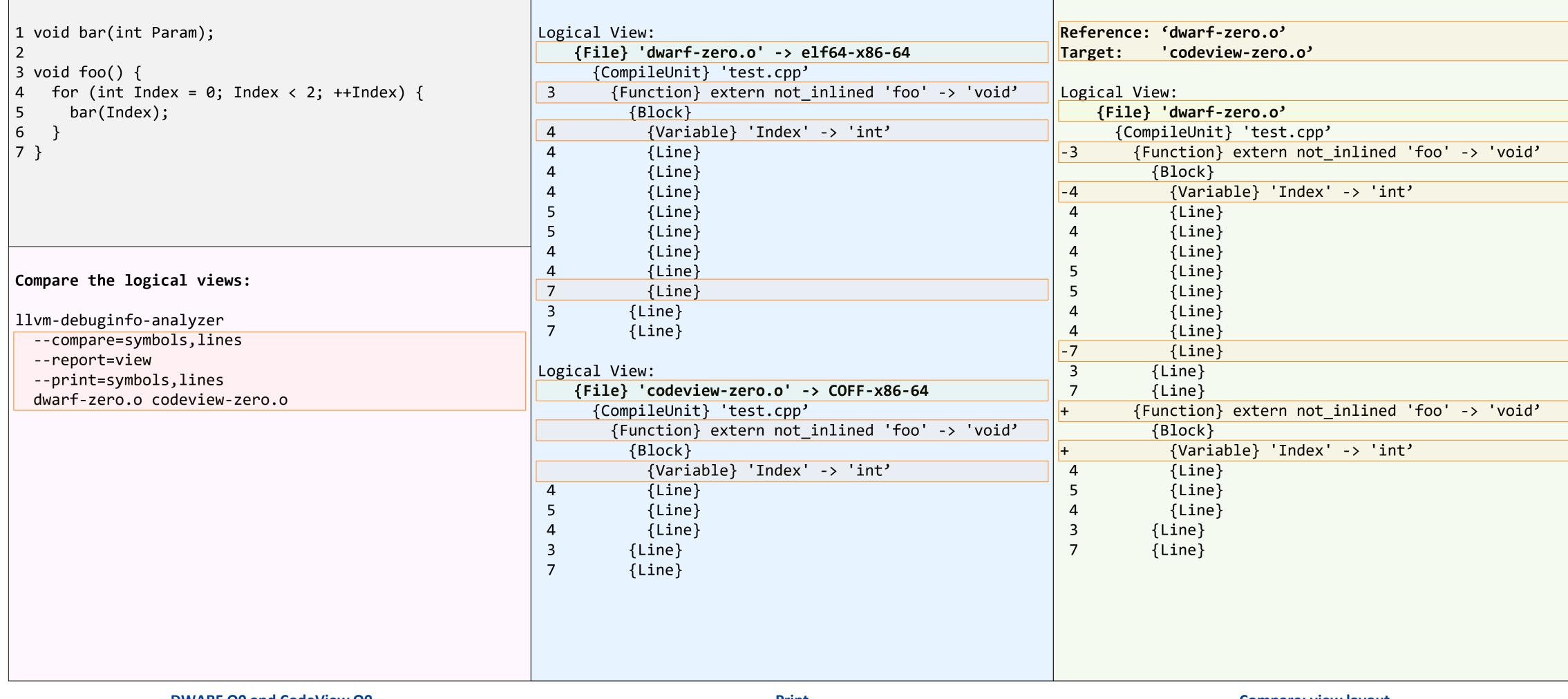


# --compare (DWARF 00 and CodeView 00) - view layout

```
1 void bar(int Param);
                                                                                                                      Reference: 'dwarf-zero.o'
                                                            Logical View:
                                                               {File} 'dwarf-zero.o' -> elf64-x86-64
                                                                                                                                  'codeview-zero.o'
                                                                                                                      Target:
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                    {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                      Logical View:
                                                                                                                           {File} 'dwarf-zero.o'
       bar(Index);
                                                                      {Block}
                                                                        {Variable} 'Index' -> 'int'
                                                                                                                             {CompileUnit} 'test.cpp'
7 }
                                                                                                                              {Function} extern not_inlined 'foo' -> 'void'
                                                                        {Line}
                                                                        {Line}
                                                                                                                                 {Block}
                                                                                                                                   {Variable} 'Index' -> 'int'
                                                                        {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                                                                                   {Line}
Compare the logical views:
                                                                        {Line}
                                                                                                                                   {Line}
                                                                                                                                   {Line}
                                                                      {Line}
 llvm-debuginfo-analyzer
                                                                      {Line}
                                                                                                                                   {Line}
  --compare=symbols, lines
                                                                                                                                   {Line}
  --report=view
                                                            Logical View:
                                                                                                                                 {Line}
  --print=symbols,lines
                                                               {File} 'codeview-zero.o' -> COFF-x86-64
                                                                                                                                 {Line}
  dwarf-zero.o codeview-zero.o
                                                                                                                               {Function} extern not_inlined 'foo' -> 'void'
                                                                 {CompileUnit} 'test.cpp'
                                                                    {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                                 {Block}
                                                                                                                                   {Variable} 'Index' -> 'int'
                                                                      {Block}
                                                                        {Variable} 'Index' -> 'int'
                                                                                                                                   {Line}
                                                                        {Line}
                                                                                                                                   {Line}
                                                                                                                                   {Line}
                                                                        {Line}
                                                                        {Line}
                                                                                                                                 {Line}
                                                                                                                                 {Line}
                                                                      {Line}
                                                                      {Line}
```

(sn) systems

# --compare (DWARF 00 and CodeView 00) - view layout



(sn) systems

#### --compare (DWARF O2 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                          Logical View:
                                                                                                                    Reference: 'dwarf-two.o'
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                               'codeview-two.o'
                                                                                                                    Target:
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                  {Function} extern not_inlined 'bar' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                    (4) Missing Scopes:
                                                                    {Parameter} '' -> 'int'
                                                                                                                       3 {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                            {CallSite} 'bar' -> 'void'
7 }
                                                                    {CallSite} 'bar' -> 'void'
                                                                                                                            {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                      1 {Function} extern not_inlined 'bar' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                    (6) Missing Symbols:
                                                                                                                        4 {Variable} 'Index' -> 'int'
                                                                    {Block}
                                                                      {Variable} 'Index' -> 'int'
                                                                                                                            {CallSiteParameter} '' -> 'void'
                                                                                                                            {CallSiteParameter} '' -> 'void'
                                                                      {Line}
Compare the logical views:
                                                                                                                            {Parameter} '' -> 'int'
                                                                      {Line}
                                                                                                                            {Parameter} '' -> 'int'
                                                                      {Line}
llvm-debuginfo-analyzer
                                                                                                                            {Parameter} '' -> 'int'
                                                                    {Line}
  --compare=symbols,lines
  --report=list
                                                          Logical View:
                                                                                                                    (1) Added Scopes:
  --print=symbols,lines
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
                                                              {File} 'codeview-two.o' -> COFF-x86-64
  dwarf-two.o codeview-two.o
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                    {Block}
                                                                      {Line}
                                                                    {Line}
```

DWARF O2 and CodeView O2 Print Compare: list layout



#### --compare (DWARF O2 and CodeView O2) - list layout

```
1 void bar(int Param);
                                                          Logical View:
                                                                                                                   Reference: 'dwarf-two.o'
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                               'codeview-two.o'
                                                                                                                   Target:
3 void foo() {
                                                                {CompileUnit} 'test.cpp'
                                                                 {Function} extern not_inlined 'bar' -> 'void'
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                                                                    (4) Missing Scopes:
                                                                    {Parameter} '' -> 'int'
                                                                                                                       3 {Function} extern not_inlined 'foo' -> 'void'
      bar(Index);
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                            {CallSite} 'bar' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
                                                                                                                            {CallSite} 'bar' -> 'void'
7 }
                                                                                                                     1 {Function} extern not_inlined 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                    (6) Missing Symbols:
                                                                                                                       4 {Variable} 'Index' -> 'int'
                                                                    {Block}
                                                                                                                           {CallSiteParameter} '' -> 'void'
                                                                      {Variable} 'Index' -> 'int'
                                                                      {Line}
                                                                                                                            {CallSiteParameter} '' -> 'void'
Compare the logical views:
                                                                                                                           {Parameter} '' -> 'int'
                                                                      {Line}
                                                                                                                            {Parameter} '' -> 'int'
                                                                      {Line}
llvm-debuginfo-analyzer
                                                                                                                            {Parameter} '' -> 'int'
                                                                    {Line}
  --compare=symbols,lines
  --report=list
                                                          Logical View:
                                                                                                                    (1) Added Scopes:
  --print=symbols,lines
                                                                                                                            {Function} extern not_inlined 'foo' -> 'void'
                                                              {File} 'codeview-two.o' -> COFF-x86-64
  dwarf-two.o codeview-two.o
                                                                {CompileUnit} 'test.cpp'
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                    {Block}
                                                                     {Line}
                                                                    {Line}
```

DWARF O2 and CodeView O2 Print Compare: list layout



#### --compare (DWARF O2 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Reference: 'dwarf-two.o'
                                                               {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                     Target:
                                                                                                                                 'codeview-two.o'
3 void foo() {
                                                                 {CompileUnit} 'test.cpp'
                                                                  {Function} extern not inlined 'bar' -> 'void'
                                                                                                                     Logical View:
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                     {Parameter} '' -> 'int'
                                                                                                                         {File} 'dwarf-two.o'
      bar(Index);
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                           {CompileUnit} 'test.cpp'
7 }
                                                                     {CallSite} 'bar' -> 'void'
                                                                                                                             {Function} extern not_inlined 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                                                                               {Parameter} '' -> 'int'
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                     {CallSite} 'bar' -> 'void'
                                                                       {CallSiteParameter} '' -> 'void'
                                                                                                                               {CallSite} 'bar' -> 'void'
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                     {Block}
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                 {Parameter} '' -> 'int'
                                                                       {Line}
                                                                                                                               {CallSite} 'bar' -> 'void'
Compare the logical views:
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                       {Line}
                                                                                                                                 {Parameter} '' -> 'int'
                                                                       {Line}
 llvm-debuginfo-analyzer
                                                                                                                               {Block}
                                                                     {Line}
  --compare=symbols, lines
                                                                                                                                 {Variable} 'Index' -> 'int'
  --report=view
                                                           Logical View:
                                                                                                                                 {Line}
   --print=symbols,lines
                                                               {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                                 {Line}
  dwarf-two.o codeview-two.o
                                                                 {CompileUnit} 'test.cpp'
                                                                                                                                 {Line}
                                                                   {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                               {Line}
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                     {Block}
                                                                       {Line}
                                                                                                                               {Block}
                                                                     {Line}
                                                                                                                                 {Line}
                                                                                                                               {Line}
```

DWARF O2 and CodeView O2 Print Compare: view layout



#### --compare (DWARF O2 and CodeView O2) - view layout

```
1 void bar(int Param);
                                                           Logical View:
                                                                                                                     Reference: 'dwarf-two.o'
                                                              {File} 'dwarf-two.o' -> elf64-x86-64
                                                                                                                     Target:
                                                                                                                                'codeview-two.o'
                                                                {CompileUnit} 'test.cpp'
3 void foo() {
                                                                  {Function} extern not_inlined 'bar' -> 'void'
                                                                                                                     Logical View:
    for (int Index = 0; Index < 2; ++Index) {</pre>
                                                                     {Parameter} '' -> 'int'
                                                                                                                         {File} 'dwarf-two.o'
      bar(Index);
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                           {CompileUnit} 'test.cpp'
                                                                                                                             {Function} extern not_inlined 'bar' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
7 }
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                               {Parameter} '' -> 'int'
                                                                                                                             {Function} extern not inlined 'foo' -> 'void'
                                                                    {CallSite} 'bar' -> 'void'
                                                                      {CallSiteParameter} '' -> 'void'
                                                                                                                               {CallSite} 'bar' -> 'void'
                                                                     {Block}
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                       {Variable} 'Index' -> 'int'
                                                                                                                                 {Parameter} '' -> 'int'
                                                                      {Line}
                                                                                                                               {CallSite} 'bar' -> 'void'
Compare the logical views:
                                                                                                                                 {CallSiteParameter} '' -> 'void'
                                                                      {Line}
                                                                                                                                 {Parameter} '' -> 'int'
                                                                      {Line}
llvm-debuginfo-analyzer
                                                                                                                               {Block}
                                                                    {Line}
  --compare=symbols,lines
                                                                                                                                 {Variable} 'Index' -> 'int'
  --report=view
                                                           Logical View:
                                                                                                                                 {Line}
  --print=symbols,lines
                                                              {File} 'codeview-two.o' -> COFF-x86-64
                                                                                                                                 {Line}
  dwarf-two.o codeview-two.o
                                                                {CompileUnit} 'test.cpp'
                                                                                                                                 {Line}
                                                                  {Function} extern not_inlined 'foo' -> 'void'
                                                                                                                               {Line}
                                                                                                                             {Function} extern not_inlined 'foo' -> 'void'
                                                                    {Block}
                                                                      {Line}
                                                                                                                               {Block}
                                                                    {Line}
                                                                                                                                {Line}
                                                                                                                               {Line}
```

DWARF O2 and CodeView O2 Print Compare: view layout



#### **Summary**

#### **Debug Information**

- Common problems
- LLVM and debug information

#### Ilvm-debuginfo-analyzer

- Command line tool that processes debug information
- Produces a uniform logical view regardless of the encoding of the debug information
- Uses a free form text output for the logical view
- Prints the logical elements representing the debug information
- Supports selection criteria to determine logical elements to print
- Finds semantic differences by comparing logical views



#### C. Future work

- Add support for binary formats:
  - WebAssembly (Wasm)
  - Extended COFF (XCOFF)
- Generate the logical views in:
  - JSON or YAML
- Process additional debug information data:
  - DWARF v5 .debug\_names section
  - CodeView public symbols stream
- Support relocatable files:
  - Ability to process objects where each function is always in a different section (deadstripping).



#### Big thank you

- Paul Robinson
- David Blaikie
- Jeremy Morse
- Stephen Tozer
- Wolfgang Pieb
- J. Ryan Stinnett
- Djordje Todorovic
- Russell Gallop
- Zequan Wu
- Michał Górny
- Kevin Athey
- Tobias Hieta
- Jonas Devlieghere

- Pavel Samolysov
- Orlando Cazalet-Hyams
- Chris Jackson
- Greg Bedwell
- Eric Christopher
- Reid Kleckner
- Alexandre Ganea
- Douglas Yung
- Nico Weber
- Fangrui Song
- Vitaly Buka
- Heejin Ahn
- Adrian Prantl



# Thank you!

