

**NAME**

**dwarf\_child**, **dwarf\_offdie**, **dwarf\_offdie\_b**, **dwarf\_siblingof**, **dwarf\_siblingof\_b** - retrieve DWARF Debugging Information Entry descriptors

**LIBRARY**

DWARF Access Library (libdwarf, -ldwarf)

**SYNOPSIS**

```
#include <libdwarf.h>
```

*int*

```
dwarf_child(Dwarf_Die die, Dwarf_Die *ret_die, Dwarf_Error *err);
```

*int*

```
dwarf_offdie(Dwarf_Debug dbg, Dwarf_Off offset, Dwarf_Die *ret_die, Dwarf_Error *err);
```

*int*

```
dwarf_offdie_b(Dwarf_Debug dbg, Dwarf_Off offset, Dwarf_Bool is_info, Dwarf_Die *ret_die,  
Dwarf_Error *err);
```

*int*

```
dwarf_siblingof(Dwarf_Debug dbg, Dwarf_Die die, Dwarf_Die *ret_die, Dwarf_Error *err);
```

*int*

```
dwarf_siblingof_b(Dwarf_Debug dbg, Dwarf_Die die, Dwarf_Die *ret_die, Dwarf_Bool is_info,  
Dwarf_Error *err);
```

**DESCRIPTION**

These functions are used to retrieve and traverse DWARF Debugging Information Entry (DIE) descriptors associated with a compilation unit. These descriptors are arranged in the form of a tree, traversable using "child" and "sibling" links; see dwarf(3) for more information. DWARF Debugging Information Entry descriptors are represented by the *Dwarf\_Die* opaque type.

Function **dwarf\_child()** retrieves the child of descriptor denoted by argument *die*, and stores it in the location pointed to by argument *ret\_die*.

Function **dwarf\_siblingof()** retrieves the sibling of the descriptor denoted by argument *die*, and stores it in the location pointed to by argument *ret\_die*. If argument *die* is NULL, the first debugging information entry descriptor for the current compilation unit will be returned. This function and function **dwarf\_child()** may be used together to traverse the tree of debugging information entry

descriptors for a compilation unit.

Function **dwarf\_siblingof\_b()** is identical to the function **dwarf\_siblingof()** except that it can retrieve the sibling descriptor from either the current compilation unit or type unit. If argument *is\_info* is non-zero, the function behaves identically to function **dwarf\_siblingof()**. If argument *is\_info* is zero, the descriptor referred by argument *die* should be associated with a debugging information entry in the type unit. The function will store the sibling of the descriptor in the location pointed to by argument *ret\_die*. If argument *is\_info* is zero and argument *die* is NULL, the first debugging information entry descriptor for the current type unit will be returned.

Function **dwarf\_offdie()** retrieves the debugging information entry descriptor at global offset *offset* in the ".debug\_info" section of the object associated with argument *dbg*. The returned descriptor is written to the location pointed to by argument *ret\_die*.

Function **dwarf\_offdie\_b()** is identical to the function **dwarf\_offdie()** except that it can retrieve the debugging information entry descriptor at global offset *offset* from either of the ".debug\_info" and ".debug\_types" sections of the object associated with argument *dbg*. If argument *is\_info* is non-zero, the function will retrieve the debugging information entry from the ".debug\_info" section, otherwise the function will retrieve the debugging information entry from the ".debug\_types" section. The returned descriptor is written to the location pointed to by argument *ret\_die*.

## Memory Management

The memory area used for the *Dwarf\_Die* descriptor returned in argument *ret\_die* is allocated by the DWARF Access Library (libdwarf, -ldwarf). Application code should use function **dwarf\_dealloc()** with the allocation type DW\_DLA\_DIE to free the memory area when the *Dwarf\_Die* descriptor is no longer needed.

## RETURN VALUES

These functions return the following values:

[DW_DLV_OK]	The call succeeded.
[DW_DLV_ERROR]	The requested operation failed. Additional information about the error encountered will be recorded in argument <i>err</i> , if it is not NULL.
[DW_DLV_NO_ENTRY]	For functions <b>dwarf_child()</b> , <b>dwarf_siblingof()</b> and <b>dwarf_siblingof_b()</b> , the descriptor denoted by argument <i>die</i> did not have a child or sibling.  For functions <b>dwarf_offdie()</b> and <b>dwarf_offdie_b()</b> , there was no debugging information entry at the offset specified by argument <i>offset</i> .

**EXAMPLES**

To retrieve the first DWARF Debugging Information Entry descriptor for the first compilation unit associated with a *Dwarf\_Debug* instance, and to traverse all its children, use:

```
Dwarf_Debug dbg;
Dwarf_Die die, die0;
Dwarf_Error de;

... allocate dbg using dwarf_init() etc ...

if (dwarf_next_cu_header(dbg, NULL, NULL, NULL, NULL, NULL, &de) !=
    DW_DLV_OK)
    errx(EXIT_FAILURE, "dwarf_next_cu_header: %s",
        dwarf_errmsg(de));

/* Get the first DIE for the current compilation unit. */
die = NULL;
if (dwarf_siblingof(dbg, die, &die0, &de) != DW_DLV_OK)
    errx(EXIT_FAILURE, "dwarf_siblingof: %s", dwarf_errmsg(de));

/* Get the first child of this DIE. */
die = die0;
if (dwarf_child(die, &die0, &de) != DW_DLV_OK)
    errx(EXIT_FAILURE, "dwarf_child: %s", dwarf_errmsg(de));

/* Get the rest of children. */
do {
    die = die0;
    if (dwarf_siblingof(dbg, die, &die0, &de) == DW_DLV_ERROR)
        errx(EXIT_FAILURE, "dwarf_siblingof: %s",
            dwarf_errmsg(de));
} while (die0 != NULL);
```

**ERRORS**

These functions may fail with the following errors:

- |                            |   |
|----------------------------|---|
| [DW_DLE_ARGUMENT]          | Arguments <i>dbg</i> , <i>die</i> or <i>ret_die</i> were NULL.  |
| [DW_DLE_DIE_NO_CU_CONTEXT] | Argument <i>dbg</i> was not associated with a compilation unit. |

[DW\_DLE\_NO\_ENTRY]

The descriptor denoted by argument *die* had no child or sibling, or there was no DWARF debugging information entry at the offset specified by argument *offset*.

**SEE ALSO**

dwarf(3), dwarf\_errmsg(3), dwarf\_get\_die\_infotypes\_flag(3), dwarf\_next\_cu\_header(3)