NAME

dwarf_object_init - allocate a DWARF debug descriptor with application-specific file access methods

LIBRARY

DWARF Access Library (libdwarf, -ldwarf)

SYNOPSIS

#include <libdwarf.h>

int
dwarf_object_init(*Dwarf_Obj_Access_Interface *iface, Dwarf_Handler errhand, Dwarf_Ptr errarg, Dwarf_Debug *dbg, Dwarf_Error *err*);

DESCRIPTION

The **dwarf_object_init**() function allocates and returns a *Dwarf_Debug* instance that uses application-supplied access methods to read file content.

The argument *iface* should point to a populated *Dwarf_Obj_Access_Interface* structure. The contents of the *Dwarf_Obj_Access_Interface* structure are described in the section *Object Access Functions* below.

The argument *errhand* should point to a function to be called in case of an error. If this argument is NULL then a default error handling scheme is used. See dwarf(3) for a description of the error handling schemes available.

The argument *errarg* will be passed to the error handler function pointed to by argument *errhand*.

The argument *dbg* should point to a memory location that will be set to a reference to the returned *Dwarf_Debug* descriptor.

The argument *err* will be used to return a *Dwarf_Error* descriptor in case of an error.

Object Access Functions

The data structures used to specify object access methods are defined in <*libdwarf.h*>.

Dwarf_Obj_Access_Interface

This structure bundles together a set of file access methods along with a pointer to application-private state.

```
typedef struct {
        void *object;
```

```
        const Dwarf_Obj_Access_Methods *methods;
} Dwarf_Obj_Access_Interface;
```

object This field points to application-specific state that will be passed as the first parameter to the actual access object methods.

methods This structure contains pointers to the functions implementing the access methods, as described below.

Dwarf_Obj_Access_Methods

This structure specifies the functions implementing low-level access.

```
typedef struct {
        int (*get_section_info)(void *obj, Dwarf_Half index,
            Dwarf_Obj_Access_Section *ret, int *error);
        Dwarf_Endianness (*get_byte_order)(void *obj);
        Dwarf_Small (*get_length_size)(void *obj);
        Dwarf_Small (*get_pointer_size)(void *obj);
        Dwarf_Unsigned (*get_section_count)(void *obj);
        int (*load_section)(void *obj, Dwarf_Half ndx,
            Dwarf_Small **ret_data, int *error);
} Dwarf_Obj_Access_Methods;
```

get_byte_order This function should return the endianness of the DWARF object by returning one of the constants DW_OBJECT_MSB or DW_OBJECT_LSB.

get_length_size This function should return the number of bytes needed to represent a DWARF offset in the object being debugged.

get_pointer_size This function should return the size in bytes, in the object being debugged, of a memory address.

get_section_count This function should return the number of sections in the object being debugged.

get_section_info This function should return information about the section at the index *ndx* by filling in the structure of type *Dwarf_Obj_Access_Section* pointed to by argument *ret*. The *Dwarf_Obj_Access_Section* structure is described below.

load_section This function should load the section specified by argument *ndx* into memory and place a pointer to the section's data into the location pointed to by argument *ret_data*.

The argument *obj* passed to these functions will be set to the pointer value in the *object* field of the associated *Dwarf_Obj_Access_Interface* structure.

The argument *error* is used to return an error code in case of an error.

Dwarf_Obj_Access_Section

This structure describes the layout of a section in the DWARF object.

```
typedef struct {
        Dwarf_Addr addr;
        Dwarf_Unsigned size;
        const char *name;
} Dwarf_Obj_Access_Section;
```

addr A pointer to the start of the section's data.
size The size of the section in bytes.
name A pointer to a NUL-terminated string containing the name of the section.

RETURN VALUES

On success, the **dwarf_object_init**() function returns DW_DLV_OK. In case of an error, the function returns DW_DLV_ERROR and sets the argument *err*.

ERRORS

The **dwarf_object_init**() function may fail with the following errors:

[DW_DLE_ARGUMENT]	One of the arguments *iface* or *dbg* was NULL.
[DW_DLE_DEBUG_INFO_NULL]	The underlying object did not contain debugging information.
[DW_DLE_MEMORY]	An out of memory condition was encountered during the execution of the function.

SEE ALSO

dwarf(3), dwarf_init(3), dwarf_init_elf(3), dwarf_object_finish(3)