

**NAME**

**archive\_entry\_clear**, **archive\_entry\_clone**, **archive\_entry\_free**, **archive\_entry\_new** - functions for managing archive entry descriptions

**LIBRARY**

Streaming Archive Library (libarchive, -larchive)

**SYNOPSIS**

```
#include <archive_entry.h>
```

```
struct archive_entry *
```

```
archive_entry_clear(struct archive_entry *);
```

```
struct archive_entry *
```

```
archive_entry_clone(struct archive_entry *);
```

```
void
```

```
archive_entry_free(struct archive_entry *);
```

```
struct archive_entry *
```

```
archive_entry_new(void);
```

**DESCRIPTION**

These functions create and manipulate data objects that represent entries within an archive. You can think of a struct `archive_entry` as a heavy-duty version of struct `stat`: it includes everything from struct `stat` plus associated pathname, textual group and user names, etc. These objects are used by `libarchive(3)` to represent the metadata associated with a particular entry in an archive.

**Create and Destroy**

There are functions to allocate, destroy, clear, and copy *archive\_entry* objects:

**archive\_entry\_clear()**

Erases the object, resetting all internal fields to the same state as a newly-created object. This is provided to allow you to quickly recycle objects without thrashing the heap.

**archive\_entry\_clone()**

A deep copy operation; all text fields are duplicated.

**archive\_entry\_free()**

Releases the struct `archive_entry` object.

**archive\_entry\_new()**

Allocate and return a blank struct `archive_entry` object.

## Function groups

Due to high number of functions, the accessor functions can be found in man pages grouped by the purpose.

`archive_entry_acl(3)` Access Control List manipulation

`archive_entry_paths(3)`  
Path name manipulation

`archive_entry_perms(3)`  
User, group and mode manipulation

`archive_entry_stat(3)` Functions not in the other groups and copying to/from *struct stat*.

`archive_entry_time(3)` Time field manipulation

Most of the functions set or read entries in an object. Such functions have one of the following forms:

### **archive\_entry\_set\_XXXX()**

Stores the provided data in the object. In particular, for strings, the pointer is stored, not the referenced string.

### **archive\_entry\_copy\_XXXX()**

As above, except that the referenced data is copied into the object.

### **archive\_entry\_XXXX()**

Returns the specified data. In the case of strings, a const-qualified pointer to the string is returned.

String data can be set or accessed as wide character strings or normal *char* strings. The functions that use wide character strings are suffixed with `_w`. Note that these are different representations of the same data: For example, if you store a narrow string and read the corresponding wide string, the object will transparently convert formats using the current locale. Similarly, if you store a wide string and then store a narrow string for the same data, the previously-set wide string will be discarded in favor of the new data.

## SEE ALSO

`archive_entry_acl(3)`, `archive_entry_paths(3)`, `archive_entry_perms(3)`, `archive_entry_time(3)`, `libarchive(3)`

## HISTORY

The **libarchive** library first appeared in FreeBSD 5.3.

## AUTHORS

The **libarchive** library was written by Tim Kientzle <kientzle@acm.org>.