NAME

archive_read_extract, **archive_read_extract2**, **archive_read_extract_set_progress_callback** — functions for reading streaming archives

LIBRARY

Streaming Archive Library (libarchive, -larchive)

SYNOPSIS

#include <archive.h>

int
archive_read_extract(*struct archive *,struct archive_entry *,int flags*);

int
archive_read_extract2(*struct archive *src,struct archive_entry *,*
    struct archive *dest);

void
archive_read_extract_set_progress_callback(*struct archive *,*
    void (*func)(void *),void *user_data);

DESCRIPTION

archive_read_extract(), **archive_read_extract_set_skip_file**()
> A convenience function that wraps the corresponding archive_write_disk(3) interfaces. The first call to **archive_read_extract**() creates a restore object using archive_write_disk_new(3) and archive_write_disk_set_standard_lookup(3), then transparently invokes archive_write_disk_set_options(3), archive_write_header(3), archive_write_data(3), and archive_write_finish_entry(3) to create the entry on disk and copy data into it. The *flags* argument is passed unmodified to archive_write_disk_set_options(3).

archive_read_extract2()
> This is another version of **archive_read_extract**() that allows you to provide your own restore object. In particular, this allows you to override the standard lookup functions using archive_write_disk_set_group_lookup(3), and archive_write_disk_set_user_lookup(3). Note that **archive_read_extract2**() does not accept a *flags* argument; you should use **archive_write_disk_set_options**() to set the restore options yourself.

archive_read_extract_set_progress_callback()
> Sets a pointer to a user-defined callback that can be used for updating progress displays during extraction. The progress function will be invoked during the extraction of large regular files. The progress function will be invoked with the pointer provided to this call. Generally, the data pointed to should include a reference to the archive object and the archive_entry object so that various statistics can be retrieved for the progress display.

RETURN VALUES

Most functions return zero on success, non-zero on error. The possible return codes include: **ARCHIVE_OK** (the operation succeeded), **ARCHIVE_WARN** (the operation succeeded but a non-critical error was encountered), **ARCHIVE_EOF** (end-of-archive was encountered), **ARCHIVE_RETRY** (the operation failed but can be retried), and **ARCHIVE_FATAL** (there was a fatal error; the archive should be closed immediately).

ERRORS

Detailed error codes and textual descriptions are available from the **archive_errno**() and **archive_error_string**() functions.

SEE ALSO

tar(1), libarchive(3), archive_read(3), archive_read_data(3), archive_read_filter(3), archive_read_format(3), archive_read_open(3), archive_read_set_options(3), archive_util(3), tar(5)