

**NAME**

<code>archive_write_add_filter_b64encode,</code>	<code>archive_write_add_filter_by_name,</code>
<code>archive_write_add_filter_bzip2,</code>	<code>archive_write_add_filter_compress,</code>
<code>archive_write_add_filter_grzip,</code>	<code>archive_write_add_filter_gzip,</code>
<code>archive_write_add_filter_lrzip,</code>	<code>archive_write_add_filter_lz4,</code>
<code>archive_write_add_filter_lzip,</code>	<code>archive_write_add_filter_lzma,</code>
<code>archive_write_add_filter_lzop,</code>	<code>archive_write_add_filter_none,</code>
<code>archive_write_add_filter_program,</code>	<code>archive_write_add_filter_uuencode,</code>
<code>archive_write_add_filter_xz</code>	

**LIBRARY**

Streaming Archive Library (libarchive, -larchive)

**SYNOPSIS**

```
#include <archive.h>

int
archive_write_add_filter_b64encode(struct archive *);

int
archive_write_add_filter_bzip2(struct archive *);

int
archive_write_add_filter_compress(struct archive *);

int
archive_write_add_filter_grzip(struct archive *);

int
archive_write_add_filter_gzip(struct archive *);

int
archive_write_add_filter_lrzip(struct archive *);

int
archive_write_add_filter_lz4(struct archive *);

int
archive_write_add_filter_lzip(struct archive *);

int
archive_write_add_filter_lzma(struct archive *);

int
archive_write_add_filter_lzop(struct archive *);

int
archive_write_add_filter_none(struct archive *);

int
archive_write_add_filter_program(struct archive *, const char * cmd);

int
archive_write_add_filter_uuencode(struct archive *);

int
archive_write_add_filter_xz(struct archive *);
```

## DESCRIPTION

`archive_write_add_filter_bzip2()`, `archive_write_add_filter_compress()`,  
`archive_write_add_filter_grzip()`, `archive_write_add_filter_gzip()`,  
`archive_write_add_filter_lrzip()`, `archive_write_add_filter_lz4()`,  
`archive_write_add_filter_lzip()`, `archive_write_add_filter_lzma()`,  
`archive_write_add_filter_lzop()`, `archive_write_add_filter_xz()`,  
The resulting archive will be compressed as specified. Note that the compressed output is always properly blocked.

`archive_write_add_filter_b64encode()`, `archive_write_add_filter_uuencode()`,  
The output will be encoded as specified. The encoded output is always properly blocked.

`archive_write_add_filter_none()`  
This is never necessary. It is provided only for backwards compatibility.

`archive_write_add_filter_program()`  
The archive will be fed into the specified compression program. The output of that program is blocked and written to the client write callbacks.

## RETURN VALUES

These functions return **ARCHIVE\_OK** on success, or **ARCHIVE\_FATAL**.

## 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_write(3)`, `archive_write_format(3)`,  
`archive_write_set_options(3)`, `cpio(5)`, `mtree(5)`, `tar(5)`