Mailparse

Introduction

Mailparse is an extension for parsing and working with email messages. It can deal with » RFC 822 and » RFC 2045 (*MIME*) compliant messages.

Mailparse is stream based, which means that it does not keep in-memory copies of the files it processes - so it is very resource efficient when dealing with large messages.

Note
Mailparse requires the mbstring extension, and mbstring must be loaded before mailparse.

This extension has been moved to the » PECL repository and is no longer bundled with PHP as of PHP 4.2.0.

Installing/Configuring

Requirements

No external libraries are needed to build this extension.

Installation

This » PECL extension is not bundled with PHP. Information for installing this PECL extension may be found in the manual chapter titled Installation of PECL extensions. Additional information such as new releases, downloads, source files, maintainer information, and a CHANGELOG, can be located here: » http://pecl.php.net/package/mailparse.

In order to use these functions you must compile PHP with mailparse support by using the *--enable-mailparse* configure option.

Windows users will enable *php_mailparse.dll* inside of *php.ini* in order to use these functions. The DLL for this PECL extension may be downloaded from either the » PHP Downloads page or from » http://pecl4win.php.net/

Runtime Configuration

The behaviour of these functions is affected by settings in *php.ini*.

Mailparse configuration options

Name	Default	Changeable	Changelog
mailparse.def_charset	"us-ascii"	PHP_INI_ALL	Available since PHP 4.1.0. Removed in PHP 4.2.0.

For further details and definitions of the PHP_INI_* constants, see the php.ini directives.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

MAILPARSE_EXTRACT_OUTPUT (integer)

MAILPARSE_EXTRACT_STREAM (integer)

MAILPARSE_EXTRACT_RETURN (integer)

Mailparse Functions

mailparse_determine_best_xfer_encoding

mailparse_determine_best_xfer_encoding -- Gets the best way of encoding

Description

string **mailparse_determine_best_xfer_encoding** (resource `$fp`)

Figures out the best way of encoding the content read from the given file pointer.

Parameters

`fp`
A valid file pointer, which must be seek-able.

Return Values

Returns one of the character encodings supported by the mbstring module.

Examples

Example #1 - mailparse_determine_best_xfer_encoding() example

```
<?php

$fp = fopen('somemail.eml', 'r');
echo 'Best encoding: ' . mailparse_determine_best_xfer_encoding($fp);

?>
```

The above example will output something similar to:

```
Best encoding: 7bit
```

mailparse_msg_create

mailparse_msg_create -- Create a mime mail resource

Description

resource **mailparse_msg_create** (void)

Create a *MIME* mail resource.

Return Values

Returns a handle that can be used to parse a message.

See Also

- mailparse_msg_free()
- mailparse_msg_parse_file()

mailparse_msg_extract_part_file

mailparse_msg_extract_part_file -- Extracts/decodes a message section

Description

string **mailparse_msg_extract_part_file** (resource `$mimemail`, mixed `$filename` [, callback `$callbackfunc`])

Extracts/decodes a message section from the supplied filename.

The contents of the section will be decoded according to their transfer encoding - base64, quoted-printable and uuencoded text are supported.

Parameters

`mimemail`
: A valid *MIME* resource, created with mailparse_msg_create().

`filename`
: Can be a file name or a valid stream resource.

`callbackfunc`
: If set, this must be either a valid callback that will be passed the extracted section, or **NULL** to make this function return the extracted section. If not specified, the contents will be sent to "stdout".

Return Values

If *`callbackfunc`* is not **NULL** returns **TRUE** on success.

If *`callbackfunc`* is set to **NULL**, returns the extracted section as a string.

Returns **FALSE** on error.

See Also

- mailparse_msg_extract_part()
- mailparse_msg_extract_whole_part_file()

mailparse_msg_extract_part

mailparse_msg_extract_part -- Extracts/decodes a message section

Description

void **mailparse_msg_extract_part** (resource `$mimemail`, string `$msgbody` [, callback `$callbackfunc`])

Warning
This function is currently not documented; only its argument list is available.

Parameters

`mimemail`
A valid *MIME* resource.

`msgbody`

`callbackfunc`

Return Values

No value is returned.

See Also

- mailparse_msg_extract_part_file()
- mailparse_msg_extract_whole_part_file()

mailparse_msg_extract_whole_part_file

mailparse_msg_extract_whole_part_file -- Extracts a message section including headers without decoding the transfer encoding

Description

string **mailparse_msg_extract_whole_part_file** (resource `$mimemail`, string `$filename` [, callback `$callbackfunc`])

Warning
This function is currently not documented; only its argument list is available.

Parameters

`mimemail`
A valid *MIME* resource.

`filename`

`callbackfunc`

Return Values

See Also

- mailparse_msg_extract_part()
- mailparse_msg_extract_part_file()

mailparse_msg_free

mailparse_msg_free -- Frees a MIME resource

Description

bool **mailparse_msg_free** (resource `$mimemail`)

Frees a *MIME* resource.

Parameters

`mimemail`
A valid *MIME* resource allocated by mailparse_msg_create() or mailparse_msg_parse_file().

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- mailparse_msg_create()
- mailparse_msg_parse_file()

mailparse_msg_get_part_data

mailparse_msg_get_part_data -- Returns an associative array of info about the message

Description

array **mailparse_msg_get_part_data** (resource `$mimemail`)

Warning
This function is currently not documented; only its argument list is available.

Parameters

`mimemail`
A valid *MIME* resource.

mailparse_msg_get_part

mailparse_msg_get_part -- Returns a handle on a given section in a mimemessage

Description

resource **mailparse_msg_get_part** (resource `$mimemail`, string `$mimesection`)

Warning
This function is currently not documented; only its argument list is available.

Parameters

`mimemail`
A valid *MIME* resource.

`mimesection`

mailparse_msg_get_structure

mailparse_msg_get_structure -- Returns an array of mime section names in the supplied message

Description

array **mailparse_msg_get_structure** (resource `$mimemail`)

Warning
This function is currently not documented; only its argument list is available.

Parameters

`mimemail`
A valid *MIME* resource.

mailparse_msg_parse_file

mailparse_msg_parse_file -- Parses a file

Description

resource **mailparse_msg_parse_file** (string `$filename`)

Parses a file. This is the optimal way of parsing a mail file that you have on disk.

Parameters

`filename`
: Path to the file holding the message. The file is opened and streamed through the parser.

Return Values

Returns a *MIME* resource representing the structure, or **FALSE** on error.

See Also

- mailparse_msg_free()
- mailparse_msg_create()

mailparse_msg_parse

mailparse_msg_parse -- Incrementally parse data into buffer

Description

bool **mailparse_msg_parse** (resource `$mimemail`, string `$data`)

Incrementally parse data into the supplied mime mail resource.

This function allow you to stream portions of a file at a time, rather than read and parse the whole thing.

Parameters

`mimemail`
A valid *MIME* resource.

`data`

Return Values

Returns **TRUE** on success or **FALSE** on failure.

mailparse_rfc822_parse_addresses

mailparse_rfc822_parse_addresses -- Parse RFC 822 compliant addresses

Description

array **mailparse_rfc822_parse_addresses** (string `$addresses`)

Parses a » RFC 822 compliant recipient list, such as that found in the *To:* header.

Parameters

`addresses`
A string containing addresses, like in: *Wez Furlong <wez@example.com>, doe@example.com*

Note
This string must not include the header name.

Return Values

Returns an array of associative arrays with the following keys for each recipient:

display	The recipient name, for display purpose. If this part is not set for a recipient, this key will hold the same value as *address*.
address	The email address
is_group	**TRUE** if the recipient is a newsgroup, **FALSE** otherwise.

Examples

Example #2 - mailparse_rfc822_parse_addresses() **example**

```
<?php

$to = 'Wez Furlong <wez@example.com>, doe@example.com';
var_dump(mailparse_rfc822_parse_addresses($to));
```

```
?>
```

The above example will output:

```
array(2) {
 [0]=>
 array(3) {
   ["display"]=>
   string(11) "Wez Furlong"
   ["address"]=>
   string(15) "wez@example.com"
   ["is_group"]=>
   bool(false)
 }
 [1]=>
 array(3) {
   ["display"]=>
   string(15) "doe@example.com"
   ["address"]=>
   string(15) "doe@example.com"
   ["is_group"]=>
   bool(false)
 }
}
```

mailparse_stream_encode

mailparse_stream_encode -- Streams data from source file pointer, apply encoding and write to destfp

Description

bool **mailparse_stream_encode** (resource `$sourcefp`, resource `$destfp`, string `$encoding`)

Streams data from the source file pointer, apply *`encoding`* and write to the destination file pointer.

Parameters

`sourcefp`
: A valid file handle. The file is streamed through the parser.

`destfp`
: The destination file handle in which the encoded data will be written.

`encoding`
: One of the character encodings supported by the mbstring module.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #3 - mailparse_stream_encode() example

```
<?php

// email.eml contents: hello, this is some text=hello.
$fp = fopen('email.eml', 'r');

$dest = tmpfile();

mailparse_stream_encode($fp, $dest, "quoted-printable");

rewind($dest);

// Display new file contents
fpassthru($dest);

?>
```

The above example will output:

```
hello, this is some text=3Dhello.
```

mailparse_uudecode_all

mailparse_uudecode_all -- Scans the data from fp and extract each embedded uuencoded file

Description

array **mailparse_uudecode_all** (resource `$fp`)

Scans the data from the given file pointer and extract each embedded uuencoded file into a temporary file.

Parameters

`fp`
A valid file pointer.

Return Values

Returns an array of associative arrays listing filename information.

filename	Path to the temporary file name created
origfilename	The original filename, for uuencoded parts only

The first filename entry is the message body. The next entries are the decoded uuencoded files.

Examples

Example #4 - mailparse_uudecode_all() **example**

```
<?php

$text = <<<EOD
To: fred@example.com

hello, this is some text hello.
blah blah blah.

begin 644 test.txt
/=&AI<R!I<R!A('1E<W0*
`
end
```

```
EOD;

$fp = tmpfile();
fwrite($fp, $text);

$data = mailparse_uudecode_all($fp);

echo "BODY\n";
readfile($data[0]["filename"]);
echo "UUE ({$data[1]['origfilename']})\n";
readfile($data[1]["filename"]);

// Clean up
unlink($data[0]["filename"]);
unlink($data[1]["filename"]);

?>
```

The above example will output:

```
BODY
To: fred@example.com

hello, this is some text hello.
blah blah blah.

UUE (test.txt)
this is a test
```