Capítulo 17: Manejo de Mail en PHP

Conexión a un server IMAP o POP3:

mail_handler=imap_open(string_mbox,user,password);

Donde mbox es de la forma:

{IP:PORT}MailBox

Ejemplos:

$mail=imap_open(“{190.190.190.190:143}INBOX”,”user”,”pass”);

Conexión a la carpeta INBOX de un servidor IMAP (puerto 143)

$mail=imap_open(“{190.190.190.190:110}”,”user”,”pass”);

Conexión a la carpeta raíz de un servidor POP3 (puerto 110)

Una vez establecida la conexión la función devuelve un handler que se utiliza en el resto de las funciones para acceder a las carpetas y mails dentro de las mismas.

Ejemplo:

Example 1. imap_open() example

```
 1
 2 $mbox = imap_open ("{your.imap.host:143}", "username", "password");
 3
 4 echo "<p><h1>Mailboxes</h1>\n";
 5 $folders = imap_listmailbox ($mbox, "{your.imap.host:143}", "*");
 6
 7 if ($folders == false) {
 8     echo "Call failed<br>\n";
 9 } else {
10     while (list ($key, $val) = each ($folders)) {
11         echo $val."<br>\n";
12     }
13 }
14
15 echo "<p><h1>Headers in INBOX</h1>\n";
16 $headers = imap_headers ($mbox);
17
18 if ($headers == false) {
19     echo "Call failed<br>\n";
20 } else {
21     while (list ($key,$val) = each ($headers)) {
22         echo $val."<br>\n";
23     }
24 }
25
26 imap_close($mbox);
27
```

Una vez terminada la conexión se usa:

imap_close(mail_handler);

Manejo de MailBoxes:

int=imap_createmailbox (mail_handler string_mbox)

String mbox debe estar codificado con imap_utf7_encode() y el formato del string es el mismo que en imap_open.

Ejemplo:

Example 1. imap_createmailbox() example

```
 1
 2 $mbox = imap_open("{your.imap.host}","username","password",OP_HALFOPEN)
 3      || die("can't connect: ".imap_last_error());
 4
 5 $name1 = "phpnewbox";
 6 $name2 = imap_utf7_encode("phpnewböx");
 7
 8 $newname = $name1;
 9
10 echo "Newname will be '$name1'<br>\n";
11
12 # we will now create a new mailbox "phptestbox" in your inbox folder,
13 # check its status after creation and finaly remove it to restore
14 # your inbox to its initial state
15 if(@imap_createmailbox($mbox,imap_utf7_encode("{your.imap.host}INBOX.$newname")))   {
16   $status = @imap_status($mbox,"{your.imap.host}INBOX.$newname",SA_ALL);
17   if($status) {
18     print("your new mailbox '$name1' has the following status:<br>\n");
19     print("Messages:   ". $status->messages   )."<br>\n";
20     print("Recent:     ". $status->recent     )."<br>\n";
21     print("Unseen:     ". $status->unseen     )."<br>\n";
22     print("UIDnext:    ". $status->uidnext    )."<br>\n";
23     print("UIDvalidity:". $status->uidvalidity)."<br>\n";
24
25
if(imap_renamemailbox($mbox,"{your.imap.host}INBOX.$newname","{your.imap.host}INBOX.$name2")) {
26       echo "renamed new mailbox from '$name1' to '$name2'<br>\n";
27       $newname=$name2;
28     } else {
29       print "imap_renamemailbox on new mailbox failed: ".imap_last_error()."<br>\n";
30     }
31   } else {
32     print  "imap_status on new mailbox failed: ".imap_last_error()."<br>\n";
33   }
34   if(@imap_deletemailbox($mbox,"{your.imap.host}INBOX.$newname")) {
35     print "new mailbox removed to restore initial state<br>\n";
36   } else {
37                         print         "imap_deletemailbox     on     new     mailbox     failed:
".implode("<br>\n",imap_errors())."<br>\n";
38   }
39
40 } else {
41   print  "could not create new mailbox: ".implode("<br>\n",imap_errors())."<br>\n";
42 }
43
44 imap_close($mbox);
45
```

Devuelve true si pudo crear el mailbox o false en caso contrario.

int=imap_deletemailbox (mail_handler, string_mbox);

Elimina el mailbox indicado, el formato de mbox es el mismo que en imap_open.

int=imap_renamemailbox (mail_handler, string_old_mbox, string_new_mbox)

Permite renombrar un mailbox, el nombre del mailbox debe estar en el mismo formato que en imap_open.

obj_array=imap_getmailboxes (mail_stream, string_ref, string_pattern)

Devuelve un vector de objetos con información sobre los mailboxes

Los objetos que se encuentran en el vector tienen seteados los siguientes data_members:

- name – Nombre del mailbox (completo) encodeado, decodificar con imap_utf7_decode()
- delimiter – Delimitador usado para separar la jerarquía de mailboxes
- attributes – Es un bitmask que puede compararse con:
  - LATT_NOINFERIORS (el mailbox no tiene subcarpetas)
  - LATT_NOSELECT (es un mailbox no seleccionable)
  - LATT_MARKED (mailbox marcado)
  - LATT_UNMARKED (mailbox no marcado)

Ejemplo:

Example 1. imap_getmailboxes() example

```
 1
 2 $mbox = imap_open("{your.imap.host}","username","password")
 3       || die("can't connect: ".imap_last_error());
 4
 5 $list = imap_getmailboxes($mbox,"{your.imap.host}","*");
 6 if(is_array($list)) {
 7   reset($list);
 8   while (list($key, $val) = each($list))
 9   {
10     print "($key) ";
11     print imap_utf7_decode($val->name).",";
12     print "'".$val->delimiter."',";
13     print $val->attributes."<br>\n";
14   }
15 } else
16   print "imap_getmailboxes failed: ".imap_last_error()."\n";
17
18 imap_close($mbox);
19
```

object=imap_status (mail_handler, string_mailbox, SA_ALL)

SA_ALL es una constante para recuperar toda la información sobre el mailbox, devuelve un objeto con los siguientes data members seteados:

- messages – número de mensajes en el mailbox
- recent – número de mensajes recientes en el mailbox
- unseen – número de mensajes no vistos en el mailbox

Ejemplo:

Example 1. imap_status() example

```
  1
  2 $mbox = imap_open("{your.imap.host}","username","password",OP_HALFOPEN)
  3       || die("can't connect: ".imap_last_error());
  4
  5 $status = imap_status($mbox,"{your.imap.host}INBOX",SA_ALL);
  6 if($status) {
  7   print("Messages:   ". $status->messages   )."<br>\n";
  8   print("Recent:     ". $status->recent     )."<br>\n";
  9   print("Unseen:     ". $status->unseen     )."<br>\n";
 10   print("UIDnext:    ". $status->uidnext    )."<br>\n";
 11   print("UIDvalidity:". $status->uidvalidity)."<br>\n";
 12 } else
 13   print "imap_status failed: ".imap_lasterror()."\n";
 14
 15 imap_close($mbox);
 16
```

int imap_num_msg (mail_handler)

Devuelve el número de mensajes en el mailbox actual. (El abierto por el mail_handler)

int imap_num_recent (mail_handler)

Devuelve el número de mensajes recientes del mailbox correspondiente a mail_handler.

Manejo de mensajes:

object=imap_fetchstructure (mail_handler, int msg_number)

Devuelve un objeto con la estructura del mensaje recuperado:

Table 1. Returned Objects for imap_fetchstructure()

type	Primary body type
encoding	Body transfer encoding
ifsubtype	True if there is a subtype string
subtype	MIME subtype
ifdescription	True if there is a description string
description	Content description string
ifid	True if there is an identification string
id	Identification string
lines	Number of lines
bytes	Number of bytes
ifdisposition	True if there is a disposition string
disposition	Disposition string
ifdparameters	True if the dparameters array exists
dparameters	Disposition parameter array
ifparameters	True if the parameters array exists
parameters	MIME parameters array
parts	Array of objects describing each message part

Cuando el mensaje es multipart “parts” es un vector donde cada elemento es un objeto con los siguientes datamembers:

- type
- encoding
- subtype
- description
- lines
- disposition

Luego según el trasnfer encoding (ver tabla 3) se puede usar la función de decodificación apropiada

Table 2. Primary body type

0	text
1	multipart
2	message
3	application
4	audio
5	image
6	video
7	other

Table 3. Transfer encodings

0	7BIT
1	8BIT
2	BINARY
3	BASE64
4	QUOTED-PRINTABLE
5	OTHER

Las funciones de decodificación provistas son:

string=imap_base64(string) convierte de base 64 a 8 bits
string=imap_8bit(string) convierte de 8 bits a quoted printable
string=imap_utf7_decode(string) convierte de 7 bits a 8 bits
string=imap_qprint(string) convierte de quoted printable a 8 bits
string=imap_binary(string) convierte de 8 bits a base64

El formato de salida “string” es 8 bits, si el formato de encoding es otro basta con usar la función apropiada.

string=imap_fetchbody (mail_handler, int msg_number, string part_number)

Recupera la parte indicada del body de un determinado mensaje. No realiza ningún tipo de decodificación.

array= imap_headers (mail_handlers)

Devuelve un vector de headers para el mailbox actual (cada header es un string y es un elemento del vector)

object=imap_rfc822_parse_headers(string headers)

Parsea un header de acuerdo a rfc822, devuelve un objeto con los siguientes data_members:

- remail
- date
- Date
- subject
- Subject
- in_reply_to
- message_id
- newsgroups
- followup_to
- references

string imap_body (mail_handler, int msg_number)

Devuelve el body de un determinado mensaje.

Envío de mail:

Enviar mail desde PHP es sencillo con la función:

bool= mail (string to, string subject, string message [, string additional_headers])

Example 1. Sending mail.

```
  1
  2 mail("rasmus@lerdorf.on.ca", "My Subject", "Line 1\nLine 2\nLine 3");
  3
```

Example 2. Sending mail with extra headers.

```
  1
  2 mail("nobody@aol.com", "the subject", $message,
   3       "From: webmaster@$SERVER_NAME\nReply-To: webmaster@$SERVER_NAME\nX-
Mailer: PHP/" . phpversion());
  4
```