Enviar emails con adjunto utilizando PHP

A la hora de desarrollar aplicaciones web, es muy frecuente encontrarse con la necesidad de programar alguna función que se encargue de enviar emails al usuario. En el caso del lenguaje PHP, disponemos de varias formas de hacerlo, siendo la más conocida el uso de la función "**mail()**", aunque por tema de seguridad, muchos proveedores de alojamiento web la tienen desactivada. En estos casos, hay que buscar alternativas como pueden ser el caso del uso de la clase PEAR Mail o bien el uso de la librería PHPMailer. Si estáis interesados en saber cómo hacer el envío de adjuntos con estos sistemas, seguid leyendo porque os lo explicamos en nuestro White Paper.

Envío de emails utilizando PEAR Mail

La clase PEAR Mail es una de las que podemos utilizar para realizar el envío de correos electrónicos. La desventaja es que tiene que estar instalada en el servidor, pero la mayoría de los proveedores de hoy en día la tienen habilitada. Una de sus principales características es que ofrece soporte de extensiones MIME, lo que permite enviar HTML con imágenes, vídeo, archivos adjuntos… Veamos a continuación un ejemplo de cómo utilizar esta clase para realizar envíos de archivos adjuntos.

```
require("Mail.php");
require("Mail/mime.php");

$to="micorreo@midominio.com";
$from = "info@hostalia.com";
$host     = 'smtp.ejemplo.com';
$username = 'origen@ejemplo.com';
$password = 'P4ssw0rd';

$subject="Prueba envío de correos con adjuntos con PEAR::Mail";
$mensajeTexto="Aquí iría el texto del cuerpo del mensaje";
$adjunto = "fichero_adjunto.zip";

$headers=array();
$headers['From']=$from;
$headers['To']=$to;
$headers['Subject']=$subject;
$headers['Return-Path']="no-reply@midominio.com";

$message = new Mail_mime();
$message->setTXTBody($messageTEXT);
$message->addAttachment($adjunto, mime_content_type($adjunto));

$mimeparams=array();
$mimeparams['text_encoding']="7bit";
$mimeparams['text_charset']="UTF-8";

$body = $message->get($mimeparams);

$headers = $message->headers($headers);

$smtp = Mail::factory('smtp',
    array ('host' => $host,
           'auth' => true,
           'username' => $username,
           'password' => $password));
$mail=$smtp->send($to, $headers, $body);
```

```
if (PEAR::isError($mail)) {
    print "Error";
}else{
    print "Correo enviado";
}
```

Una vez que hemos visto todo el código, vamos a empezar a explicar lo que significa cada una de sus líneas.

Lo primero que debemos hacer es indicar las clases que vamos a utilizar para realizar los envíos de nuestros correos electrónicos.

```
require("Mail.php");
require("Mail/mime.php");
```

A continuación lo que hacemos es declarar algunas variables con datos que luego utilizaremos para realizar el envío de los correos electrónicos.

```
$to= "micorreo@midominio.com";
$from = "info@hostalia.com";
$host     = 'smtp.ejemplo.com';
$username = 'origen@ejemplo.com';
$password = 'P4ssw0rd';
$subject="Prueba envío de correos con adjuntos con PEAR::Mail";
$mensajeTexto="Aquí iría el texto del cuerpo del mensaje";
$adjunto = "fichero_adjunto.zip";
```

- **$to**: Se indica la dirección a la que se enviará el email.
- **$from**: Dirección quien envía el correo.
- **$host**: Nombre del servidor desde el que hacemos el envío del correo electrónico.
- **$username**: Usuario de la cuenta de correo para realizar un envío autentificado.
- **$password**: Contraseña del usuario del correo electrónico.
- **$subject**: Asunto que aparecerá en el correo enviado.
- **$mensajeTexto**: Texto que irá dentro del correo electrónico.
- **$adjunto:** Archivo que adjuntaremos en nuestro correo.

Lo siguiente será declarar las cabeceras del correo, necesarias para que el envío se haga con éxito y no dé ningún tipo de error. Aunque hay varios tipos de cabeceras, es suficiente con indicar quién envía el correo, quién lo recibe, el asunto y la dirección de correo electrónico de respuesta.

```
$headers=array();
$headers['From']=$from;
$headers['To']=$to;
$headers['Subject']=$subject;
$headers['Return-Path']="no-reply@midominio.com";
```

Una vez que tenemos todo lo anterior es hora de empezar a utilizar la extensión **PEAR Mail** para realizar el envío de los emails. Para ello, lo primero será crearnos nuestro objeto con el que trabajaremos.

```
$message = new Mail_mime();
```

Mediante el uso del método "**setTXTBody**" del objeto creado, le indicaremos el texto que queremos enviar y que lo hemos almacenado en la variable "**$mensajeTexto**".

```
$message->setTXTBody($messageTEXT);
```

Utilizaremos también su método "**addAttachment**" para indicar el archivo adjunto que vamos a enviar. A este método se le pasa dos parámetros. El primero de ellos es la ubicación del archivo, que tenemos almacenada en la variable "**$adjunto**", mientras que como segundo parámetro se le pasará el tipo de archivo a adjuntar. Si el tipo de archivo varía, podemos utilizar la función de PHP "**mime_content_type**" que nos devolverá el tipo de archivo.

```
$message->addAttachment($adjunto, mime_content_type($adjunto));
```

También será necesario indicar el tipo de codificación que utilizaremos en el correo. En nuestro caso un encoding de 7 bits y formato UTF-8.

```
$mimeparams=array();
$mimeparams['text_encoding']="7bit";
$mimeparams['text_charset']="UTF-8";
```

Para obtener el cuerpo del mensaje con la codificación correcta, haremos la llamada al método "**get**" del objeto Mail que hemos creado y al que se le pasará los parámetros "**Mime**" que hemos definido anteriormente. Todo esto lo almacenamos en una variable de nombre "**$body**".

```
$body = $message->get($mimeparams);
```

De forma parecida, actuaremos con las cabeceras, pero en este caso haciendo uso del método "**headers**".

```
$headers = $message->headers($headers);
```

Utilizaremos el "**host**", el "**username**" y el "**password**" que hemos definido al principio para crear un objeto "**smtp**" que nos permitirá realizar el envío de toda la información.

```
$smtp = Mail::factory('smtp',
    array ('host' => $host,
           'auth' => true,
           'username' => $username,
           'password' => $password));
```

Por último, hacemos uso del método "**send**" del objeto que hemos creado anteriormente para realiza el envío del correo electrónico.

```
$mail=$smtp->send($to, $headers, $body);
```

Si todo ha ido bien, el correo electrónico llegará a su destinatario sin problemas.

Envío de emails utilizando PHP Mailer

La otra opción que hoy os queremos explicar a la hora de poder enviar correos electrónicos con archivos adjuntos desde nuestra programación, es mediante el uso de la librería **PHP Mailer**. Se trata de una clase cuyo principal objetivo es facilitar al desarrollador la tarea del envío de correos electrónicos desde la propia página web.

Veamos primero el código completo y luego explicaremos paso a paso cada una de sus líneas.

```
require('PHPMailer/class.phpmailer.php');
require('PHPMailer/class.smtp.php');

$mail = new PHPMailer();
$mail->IsSMTP();
$mail->SMTPAuth = true;
$mail->Host = "smtp.ejemplo.com";
$mail->Username = "origen@ejemplo.com";
$mail->Password = "P4ssw0rd";
$mail->From = "info@hostalia.com";
$mail->AddAddress("micorreo@midominio.com");
$mail->Subject = "Envío de correo utilizando PHPMailer";
$mail->Body = "Aquí iría el texto del cuerpo del mensaje";
$mail->AddAttachment("fichero_adjunto.zip","fichero_adjunto.zip");
$mail->Send();
```

Lo primero de todo será añadir a nuestro proyecto las librerías necesarias.

```
require('PHPMailer/class.phpmailer.php');
require('PHPMailer/class.smtp.php');
```

Lo siguiente que hacemos es instanciar un objeto de la clase.

```
$mail = new PHPMailer();
```

Indicamos que vamos hacer el envío utilizando SMTP y que además será autenticado mediante usuario y contraseña.

```
$mail->IsSMTP();
$mail->SMTPAuth = true;
```

A continuación, indicamos los datos de configuración del servidor SMTP para realizar el envío del correo.

```
$mail->Host = "smtp.ejemplo.com";
$mail->Username = "origen@ejemplo.com";
$mail->Password = "P4ssw0rd";
```

También tendremos que indicar la dirección del correo que hace el envío.

```
$mail->From = "info@hostalia.com";
```

Así como la dirección de destino. En nuestro ejemplo lo hemos puesto fijo pero podría ser perfectamente un valor que se obtuviera de un formulario de contacto.

```
$mail->AddAddress("micorreo@midominio.com");
```

Indicamos el asunto y el texto del cuerpo del mensaje.

```
$mail->Subject = "Envío de correo utilizando PHPMailer";
$mail->Body = "Aquí iría el texto del cuerpo del mensaje";
```

Y por último, antes de hacer el envío, adjuntamos el documento mediante la función "**AddAttachment**". Este método recibe dos parámetros. El primero de ellos corresponde con la ubicación del archivo adjunto que vamos a enviar, mientras que el segundo, es el nombre que queremos que tenga el archivo adjunto que recibirá el destinatario.

```
$mail->AddAttachment("fichero_adjunto.zip","fichero_adjunto.zip");
```

Por último, solo falta enviar el correo.

```
$mail->Send();
```

Como hemos visto a lo largo de este White Paper, que no podamos utilizar la función "**mail()**" de PHP, no es motivo para no poder enviar correos electrónicos mediante programación. Lo único que deberemos hacer, es utilizar alguna librería alternativa como las que hoy hemos visto.