

Tutorial de phpMailer

Este tutorial describe cómo utilizar phpMailer y explica el uso de las funciones de la clase.

Por Tom Klingenberg

Contenido

1. [phpMailer](#)
¿Qué es? ¿Qué hace? ¿Quién lo necesita? – Un resumen de las características de la clase.
2. [La primera vez](#)
El primer correo con phpMailer. Vea como empezar. Si usted ha analizado completamente el código de la clase puede saltarse este capítulo.
3. [Enviando archivos adjuntos](#)
Como enviar correos con archivos adjuntos.
4. [Enviando mensajes HTML](#)
Utilice las características de HTML integradas en la clase.
5. [¿Dónde puedo conseguir ayuda o soporte?](#)
Sitios de interés: la web de phpMailer, listas de correo, etc..
6. [Otros recursos de correo electrónico y PHP](#)
Hay muchísima información en Internet sobre estos temas.

phpMailer

phpMailer es una clase de PHP que proporciona un paquete de funciones para enviar emails. Las características principales son correos en formato HTML y archivos adjuntos. La clase soporta casi todas las posibilidades, tanto antiguas como nuevas: Usted puede usar cualquier característica del protocolo SMTP: múltiples destinatarios, múltiples copias (CC, BCC), etc. En breve podrá manejar perfectamente esta clase.

Como usted sabe, puede usar la función mail() de PHP para enviar correos. Así Pues ¿para que emplear una clase? ¿No es un trabajo más engorroso y lento? Efectivamente. Pero es un precio que merece la pena pagar, ya que la función mail() no le facilita el envío de correos con formato HTML ni archivos adjuntos. La clase phpMailer le permite usar su propio servidor SMTP y todas las características de las plataformas basadas en Unix.

Este tutorial le enseña a implementar la clase en sus scripts o sitios web y le ayuda a construir su propia aplicación de correo electrónico. Si quiere enviar emails simples (solo texto plano) le sugiero usar la función mail(), ya que forma parte del propio lenguaje PHP y no necesita implementar ninguna clase.

La primera vez

Este apartado puede parecer muy simple, pero le ofrece una primera toma de contacto con la clase. Incluso si usted no ha usado nunca antes la función mail() lea este apartado, que contiene información general sobre el correo electrónico. Antes de seguir adelante asegúrese de que tiene la clase en su disco duro. Si no está seguro, puede probarlo con el siguiente script:

```
<?php
require("class.phpmailer.php");
$mail = new phpMailer();
?>
```

Guardelo como un fichero PHP en el mismo directorio en el que almacena sus scripts (donde debería tener la clase). Carguelo en el navegador. Si no se produce ningún error, la clase está copiada donde debe estar. Si obtiene algún mensaje de aviso o error lea el manual de PHP al respecto.

Este pequeño código es el comienzo de cualquier uso de phpMailer.

`require("class.phpmailer.php");` Con él se incluye la clase en el script actual (aunque usemos

require() decimos "incluir"), y `$mail = new phpMailer();` implementa el objeto `$mail` de la clase. Eso quiere decir que tiene todas sus propiedades y métodos.

Esta parte es aburrida. Vayamos a enviar nuestro primer correo. Recuerde lo que necesita: Una dirección de destino, una de remite, un asunto y un texto (el cuerpo del mensaje o, simplemente, el cuerpo). Me imagino que usted ya ha pensado en eso.

Además, necesitamos saber el método por el que se va a enviar el correo ¿Sabe de que hablo? Necesitamos un programa para comunicar con un servidor SMTP que realmente envíe el mensaje a través de La Red. En las plataformas Unix el más popular es SendMail. En cualquier caso es necesario un servidor SMTP.

Con phpMailer es posible usar varios modos de envío de correos electrónicos. Por ahora yo le sugiero SMTP.. Es importante que el servidor acepte la dirección del remitente. Si no, el envío fallará. Ya está todo preparado. Así es como funciona:

```
<?php
require("class.phpmailer.php");
$mail = new phpMailer();
$mail->IsSMTP(); // telling the class to use SMTP
$mail->Host = "smtp.email.com"; // SMTP server
$mail->From = "from@email.com";
$mail->AddAddress("myfriend@site.com");

$mail->Subject = "first mailing";
$mail->Body = "hi ! \n\n this is First mailing I made
myself with phpMailer !";
$mail->WordWrap = 50;

if(!$mail->Send())
{
    echo "Message was not sent";
    echo "Mailer Error: " . $mail->ErrorInfo;
}
else
{
    echo "Message has been sent";
}
?>
```

Grabe el listado en un script PHP y cambia los valores que necesite. A continuación se detalla como opera este código:

Make it a php file and change the values to your needs. For this, here is a list from top to bottom what this script does:

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

Esta línea establece que el modo de envío es SMTP (el más habitual).

```
$mail->Host = "smtp.email.com";
```

En esta línea se indica que el servidor SMTP es smtp.email.com. Sustituya este dato por la dirección de su servidor SMTP. Por cierto. Si lo desea, puede incluir varios servidores SMTP, separándolos con punto y coma (;), así: `"smtp.email.com;smtp2.email.com"`. Si el primero falla, se intentará con el segundo. Esto es una buena medida, ya que a veces los servidores se caen.

Escriba la dirección del remitente, que aparecerá en la cabecera del mensaje, como se ve en la siguiente línea:

```
$mail->From = "from@email.com";
```

Si no le gusta que aparezca en la cabecera de sus mensajes su dirección utilice, además, la propiedad FromName, como se ve en la siguiente línea.

```
$mail->FromName = "Your Name";
```

A continuación estableceremos la dirección de destino del mensaje. También puede indicar el nombre del destinatario. Vea, en el siguiente fragmento, ambas posibilidades:

```
$mail->AddAddress("myfriend@site.com", "Friends name");  
$mail->AddAddress("myfriend@site.com");
```

La siguiente parte es también muy sencilla. Se trata de establecer el asunto y el cuerpo del mensaje.

A continuación establecemos el ancho, en caracteres con el que se presentará el cuerpo del mensaje, así:

```
$mail->WordWrap = 50;
```

Y, de Nuevo, algo muy fácil. El envío del mensaje con la siguiente línea:

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

En este pequeño script se combina esto con un mensaje de error si ha habido algún problema.

Enviando archivos adjuntos

Ríndase a la evidencia. Enviar emails simples es aburrido, cuando usted puede adjuntar la foto de su perro o cualquier otro archivo que desee al correo electrónico. La clase phpMailer le permite hacerlo de modo muy fácil. Incluso si usted quiere adjuntar más de un archivo al mismo mensaje, no hay problema. Puede adjuntar todos los archivos que desee. Pero piénselo. El tamaño del correo aumenta con cada archivo adjunto.

No se asuste. Hay dos maneras de adjuntar un archivo: la primera, y más habitual, es desde el propio sistema de archivos. La segunda es incorporar datos binarios desde una cadena. Es lo que se conoce como Stringattachment. Esto le permite incorporar datos de una base de datos a su mensaje.

Adjuntando archivos

En su script puede colocar un Nuevo comando entre `$mail = new phpMailer(); y !$mail->Send();`. Se trata de `AddAttachment($path);`. Y eso es todo. Esta simple línea adjunta un archivo a su email. Preste especial atención al parámetro del método. Debe establecer la ruta donde se encuentra el archivo que desea adjuntar, si es diferente de aquella donde está el script.

Si quiere más opciones, o especificar tipos de codificación o mime del fichero, puede añadir otros tres parámetros opcionales, como se ve a continuación:

```
AddAttachment($path, $name, $encoding, $type);
```

`$name` es un parámetro opcional para que el fichero se reciba con un nombre determinado, en lugar del original.

`$encoding` es algo más técnico. Con este parámetro puede indicar el tipo de codificación del fichero adjunto. Por defecto es `base64`. Otros tipos posibles son: `7bit`, `8bit`, `binary` & `quoted-printable`. Si le interesa conocer detalles vea la documentación del protocolo SMTP y los diferentes tipos de codificación. En general, los servidores tienden a trabajar con una codificación y tratan de aplicarla a los ficheros adjuntos.

`$type` es el tipo mime de su fichero adjunto. En Internet usted no especifica el tipo de fichero. Cuando termina con un punto y un sufijo se usa el tipo MIME (Multipurpose Internet Mail Extensions). Este parámetro le da la posibilidad de cambiar el valor por defecto `application/octet-stream` (que funciona con todos los ficheros a un tipo más específico, como por ejemplo `image/jpeg` para imágenes jpg).

String Attachments

Stringattachments ha sido soportado por phpMailer desde la version 1.29. Este modo funciona tan bien como `AddAttachment()` y se llama `AddStringAttachment($string, $filename, $encoding, $type)`.

Por supuesto, usted debe proporcionar la cadena en el parámetro `$string`. Como la cadena se enviará como archivo adjunto, el parámetro `$filename` no es opcional.

El resto es igual que en el método [anterior](#).

Así pues, ¿por qué deberíamos usar `AddStringAttachment` en lugar de `AddAttachment`? ¿Es solo para ficheros de texto? No. En absoluto. Es válido para enviar bases de datos, por ejemplo. La información en una base de datos puede ser procesada siempre como una cadena (lo que se suele llamar un blob). Puede hacer una consulta de lectura y pasarle el resultado a una cadena.

Adjuntos integrados en el mensaje

Existe otro modo de adjuntar archivos. Si usted quiere crear un mensaje HTML con imágenes tiene que adjuntar la imagen y montar un enlace a la misma como el siguiente: ``. Por ejemplo, usted añade una imagen como adjunto integrado de, digamos, "my-Photo", que es un alias para la imagen. En el código HTML incluirá ``. Así de fácil.

Aquí tiene la forma de incluir adjuntos integrados en detalle:

```
$mail->AddEmbeddedImage($filename, $cid, $name);
```

Usando esta función con los valores del ejemplo anterior, el código es el siguiente:

```
$mail->AddEmbeddedImage('my-photo.jpg', 'my-photo', 'my-photo.jpg');
```

Para obtener más información sobre los correos en HTML vea el apartado [Enviando mensajes HTML](#).

Manejando adjuntos

Si usted quiere adjuntar múltiples ficheros o cadenas, simplemente llame a los métodos `AddAttachment()` o `AddStringAttachment()` tantas veces como necesite. Si quiere eliminar un adjunto puede especificarlo, pero también puede eliminar todos los adjuntos del correo usando el método `ClearAttachments()`. Recuerde, con él eliminará todos los adjuntos, cadenas y adjuntos integrados.

Enviando mensajes HTML

A cualquiera que tenga un cierto sentido artístico le gusta diseñar los mails que envía. Usted puede usar HTML para este propósito. Si no está familiarizado con HTML consulte algún manual.

Pero enviar emails con formato HTML no solo depende de su conocimiento de este lenguaje. También depende de que el programa cliente de correo electrónico del usuario sea compatible con el formato. Para aquellos clientes que no lo sean, usted puede escribir un cuerpo del mensaje alternativo en texto plano.

Veamos como hacerlo. En primer lugar crearemos un mensaje HTML muy sencillo:

```
<?php

require("class.phpmailer.php");
$mail = new phpMailer();
$mail->IsSMTP(); // telling the class to use SMTP
$mail->Host = "smtp.email.com"; // SMTP server

$mail->From = "from@email.com";
$mail->AddAddress("myfriend@site.com");

$mail->Subject = "look, it's a HTML message";
$mail->Body = "hi <b>my friend</b>! \n\n this message
uses html entities !";

?>
```

Es tan fácil como crear un correo normal. Simplemente asigne una cadena con los formatos HTML a `$mail->body`. Antes de enviarlo debemos decirle al objeto que hayamos creado a partir de la clase phpMailer que el mensaje está en formato HTML. Entrando en detalles esto significa que el cuerpo del mensaje recibirá el ajuste `multipart/alternative`. Esto se hace así:

```
$mail->IsHTML(true);
```

Y, para que todo quede atado usamos `$mail->AltBody="hi my friend! \n\n this message uses html entities, but you prefer plain text !";`, para establecer un cuerpo alternativo. Si lo hace así, el mensaje se autoajustará como `multipart/alternative` y no necesitará usar más `$mail->IsHTML(true);`.

Así es como puede enviar mensajes HTML. Solo aplique el método `Send()` del objeto y el mensaje será enviado. No obstante recuerde que algunos usuarios desconfían de estos mensajes, ya que han sido usados, en ocasiones, para el envío de virus. Además, ocupan más que un mensaje en texto plano. Por el otro lado, resultan más atractivos y personalizables. Además, también los mensajes con archivos adjuntos se pueden usar para enviar virus, y nadie renuncia a la posibilidad de adjuntar archivos. Depende de usted, en cada caso, decidir que tipo de mensaje va a enviar.

Si quiere enviar mensajes HTML con imágenes, animaciones de Flash u otras incrustaciones, phpMailer le permite hacerlo. La inserción de imágenes está explicada en el apartado [Adjuntos integrados](#) en detalle. Más brevemente, aquí tiene dos líneas como ejemplo:

```
$mail->AddEmbeddedImage("rocks.png", "my-attach",  
"rocks.png"));  
$mail->Body = 'Embedded Image:  Here is an image!';
```

¿Dónde puedo conseguir ayuda y soporte?

Como se dijo anteriormente, phpMailer es open source, publicado bajo una licencia GNU. Creo que la mayoría de preguntas iniciales han sido respondidas en este tutorial, pero puede haber algo de lo que no hayamos hablado y usted podría desear ayuda del autor de la clase o de la comunidad de usuarios. Para esto, phpMailer tiene su propia página en sourceforge (de donde usted, probablemente, haya descargado este material), y una lista de correo donde puede poner sus preguntas.

- Página principal de phpMailer: <http://phpmailer.sourceforge.net/>
- Lista de correo de phpMailer: <http://lists.sourceforge.net/lists/listinfo/phpmailer-general>
- Documentación de phpMailer: <http://phpmailer.sourceforge.net/phpdoc/phpmailer.html>

Otros recursos de correo electrónico y PHP

Hay mucha información en La Red sobre email, PHP y phpMailer:

Página principal de phpMailer

<http://phpmailer.sourceforge.net/>

Referencia de la function mail() de PHP

<http://www.php.net/manual/en/ref.mail.php>

o bien <http://www.php.net/mail>

EI RFC 833

<http://www.w3.org/Protocols/rfc822/>

MIME Email. EI RFC 2046

<http://www.ietf.org/rfc/rfc2046.txt>
