

TEMA 10

Envío

de

Emails

La función mail()

La función mail() es una función propia de PHP, que ya incorpora en su núcleo.

La sintaxis es:

```
mail($to, $subject, $message[, $add_headers[, $add_parameters]])
```

Más información podemos encontrar en:

<http://www.php.net/manual/es/function.mail.php>

\$to

Es un string que contiene la dirección de destino de nuestro correo electrónico.

Algunos ejemplos son:

- `usuario@example.com`
- `usuario@example.com, otrousuario@example.com`
- `Usuario <usuario@example.com>`
- `Usuario usuario@example.com, Otro Usuario
otrouusuario@example.com`

\$subject

Es un string que contiene el asunto del correo electrónico que se envía.

Algunos ejemplos son:

- "Saludos desde el Servidor de PHP"
- "Notificación importante"
- "Lo que os de la gana"

\$message

Es un string que contiene el mensaje que queremos enviar en el correo.

Cada línea del correo debería ser separada por un CRLF (`\r\n`) y no tener más de 70 caracteres. Eso es debido a que el mensaje, por defecto, se envía como texto plano.

Con los tres parámetros anteriores, ya podemos enviar un correo electrónico. La precaución que debemos tener es que no queremos que sea tratado como Spam por los gestores de correo como gmail o hotmail.

Additional_headers (opcional)

El parámetro `additional_headers` puede usarse para indicar opciones adicionales.

se usa típicamente para añadir cabeceras extra como:

- `From`, se refiere a quien envía el mensaje.
- `Cc`, para indicar usuarios en copia.
- `Bcc`, igual que `Cc` pero los usuarios van ocultos, es decir, no ven las direcciones de los demás.

Ejercicio 1

Vamos a empezar enviando un mail de texto plano.

```
<?php
    $to = "iescierva.carlos@gmail.com";
    $subject = "Ejercicio 1: Prueba de envío";
    $message = "Si lees esto es que se ha enviado
correctamente";
    mail($to, $subject, $message);
?>
```

Si ejecutamos este script, en la página web no aparece nada, pero si abrimos el correo, vemos que se ha enviado, aunque probablemente lo encontremos en la carpeta Spam.

Ejercicio 2

Veamos otro ejercicio con varias cabeceras y en formato HTML.

```
<?php
$to = 'iescierva.carlos@gmail.com';
$from = 'fake@iescierva.net';
$subject = 'Ejercicio 2: Mail con cabeceras';
$message = 'Si lees esto es que se ha enviado correctamente
este segundo mail';
$header = 'From: ' . $from . "\r\n";
$header .= 'Cc: iesbohio.carlos@gmail.com1 . "\r\n";
$header .= 'Reply-to: carlosabris@iescierva.net' . "\r\n";
$header .= 'MIME-Version: 1.0' . "\r\n";
$header .= 'Content-type: text/html; charset=utf-8' . "\r\n";
$header .= 'X-Mailer: PHP/' . phpversion();
if(mail($to, $subject, $message, $header)){
    echo "Mensaje enviado correctamente";
}else{
    echo "Fallo al enviar el mensaje";
}
?>
```


Ejercicio 2

Nos permite indicar quien envía el mensaje

```
$header = 'From: ' . $from . "\r\n";
```

Envía copia a esta otra dirección

```
$header .= 'Cc: iesbohio.carlos@gmail.com1 . "\r\n";
```

Indica la dirección de respuesta al mail

```
$header .= 'Reply-to: carlosabris@iescierva.net' . "\r\n";
```

Indica el formato del mail que se recibe

```
$header .= 'MIME-Version: 1.0' . "\r\n";
```

Indicamos que es un correo en html

```
$header .= 'Content-type: text/html; charset=utf-8' . "\r\n";
```

Envía la cabecera con la versión de PHP

```
$header .= 'X-Mailer: PHP/' . phpversion();
```

Comprobamos si todo ha ido bien o ha fallado el envío

```
if(mail($to, $subject, $message,$header)){
```

```
    echo "Mensaje enviado correctamente";
```

```
}else{
```

```
    echo "Fallo al enviar el mensaje";
```

```
}
```


Ejercicio 3

En este tercer ejercicio vamos a enviar un email a partir de una plantilla HTML, que llamaremos `plantillaemail.html`, como la siguiente:

```
<html>
<head>
</head>
<body>
  <h2 style="color:blue;">El mensaje se ha enviado con fecha:
  {{fecha}}</h2>
  <h1 style="color:red;">Bienvenido al sistema,
  <strong>{{nombre}}</strong></h1>
  <p>Gracias por confiar en nosotros</p>
</body>
</html>
```


Cont. ejercicio 3

Ahora, en nuestro script PHP, eliminamos la línea donde asignamos a `$message` el mensaje del email y añadimos, después de todas las cabeceras, las siguientes:

```
$datos_usuario = array('{{nombre}}' => 'Carlos El Profe',  
                        '{{fecha}}' => date('d/m/Y H:i:s'));  
  
$message = file_get_content('plantillaemail.html');  
  
foreach($datos_usuario as $key => $value) {  
    $message = str_replace($key, $value, $message);  
}
```

Otra mejora que dejo planteada es que todo el script se pueda definir como una función, que podríamos llamar `envioEmail`, y así la podemos tener en un archivo diferente.

Observaciones

Vemos que este método nos permite enviar correos a nuestros usuarios/clientes.

Pero hemos de llevar cuidado si queremos hacer envío masivo (como felicitar la Navidad a todos nuestros suscritos) porque entidades como Google pueden bloquear nuestros correos (por la IP de salida o el nombre del dominio) y no ser entregados. Ni esos ni ningún otro a partir de ese momento.

Para evitar que ocurra, podemos recurrir a externalizar el servicio. Hay determinadas Webs LEGALES que nos permiten enviar de forma masiva, y que incluso tienen desarrollada una API para PHP, como por ejemplo <http://www.mailchimp.com>

Librería PHP Mailer

Ahora, en lugar de usar la función nativa de PHP que hemos visto antes, vamos a evolucionar usando una API (una de las muchas que tiene PHP para casi cualquier cosa) que nos permite el envío de email de forma más sencilla.

<http://phpmailer.worxware.com>

<https://github.com/PHPMailer/PHPMailer>

Esta librería nos va a permitir hacer todo lo anterior y también cosas que no hemos visto como añadir archivos adjuntos a nuestro correo.

Instalación

Para instalar la librería PHPMailer:

- Descargar la librería desde GitHub
- Crear una carpeta en nuestro proyecto.
- Descomprimir el archivo descargado dentro de la carpeta que hemos creado.

Así conseguimos que los archivos de la API queden separados en una carpeta individual. Si tuviéramos varias librerías, cada una iría en una carpeta distinta e incluso podríamos tenerlas todas dentro de otra carpeta llamada librerías.

En nuestro ejemplo llamaremos a la carpeta 'lib'.

Uso

Para poder usar la librería en nuestro script sólo tenemos que incluir la siguiente línea:

```
require 'lib/PHPMailerAutoload.php' ;
```

A partir de ahora, hemos de tener en cuenta que la librería funciona mediante POO, lo cual implica que debemos definir un objeto de tipo PHPMailer y asignarle valores a través de sus propiedades y métodos.

Un ejemplo sería:


```
<?php
require( 'lib/PHPMailerAutoload.php' );

$mail = new PHPMailer
$mail->From = 'carlos@iescierva.net';
$mail->FromName = 'Carlos El Profe';
$mail->addAddress( 'iescierva.carlos@gmail.com' );
// $mail->addAddress( 'iesbohio.carlos@gmail.com' );
$mail->addReplyTo( 'No-Reply@iescierva.net' );
// $mail->addCC( 'cc@example.com' );
// $mail->addBCC( 'bcc@example.com' );
$mail->isHTML( true );
$mail->Subject = 'Prueba de PHPMailer';
$mail->Body = file_get_contents( 'plantillaemail.html' );
$datos_usuario = array( '{{nombre}}' => 'Carlos El Profe',
                        '{{fecha}}' => date( 'd/m/Y H:i:s' ) );
foreach( $datos_usuario as $key => $value ) {
    $mail->Body = str_replace( $key, $value, $mail->Body );
}
$mail->AltBody = "Este mensaje contiene HTML, por favor activa la opción";

if( $mail->send() ) {
    echo 'Correo enviado';
} else {
    echo 'Fallo en el envío';
    print_r( $mail->ErrorInfo );
}
?>
```


Adjuntar Archivos

Para adjuntar un archivo sólo necesitamos añadir:

```
$mail->addAttachment('miarchivo.txt');
```

Y esta línea se podría añadir tantas veces como archivos queramos envía en el mismo email.

Configurar SMTP

Si queremos enviar los correos de nuestra aplicación a través del correo propio de la aplicación (p.e. `info@midominio.com`) necesitamos configurar una serie de parámetros, que son en el caso de Gmail los siguientes:

```
$mail->isSMTP();  
$mail->Host = 'smtp.gmail.com';  
$mail->SMTPAuth = true;  
$mail->Username = 'iescierva.carlos@gmail.com';  
$mail->Password = 'micontraseña';  
$mail->SMTPSecure = 'tls';  
$mail->Port = 587;
```