IES NTRA. SRA. DE LOS REMEDIOS DE UBRIQUE

Tema 10 – Envíos de Email

2º DAW – Desarrollo Web Entorno Servidor

Profesor Juan Carlos Moreno

CURSO 2023/2024

Tabla de Contenido

10 Envíos de Email

10.1 Introducción

El envío de email desde PHP es otra de las funciones básicas de cualquier aplicación web y que tarde o temprano tendremos que incluir en alguno de nuestros proyectos.

Actualmente existen dos métodos bien diferenciados para enviar email desde PHP:

- Función mail() de PHP
- La clase PHPMailer

A continuación veremos las características de cada uno de los métodos. Pero antes hay que decir que para que funcione en un servidor localhost, se necesita tener instalado al menos un servidor de correo saliente o SMTP, tarea sencilla pero que en principio se puede complicar bastante.

Para dicha función existen distintas posibilidades:

1. Instalar fake SMTP
2. Usar un servidor SMTP

10.2 Fake SMTP

Para realizar nuestros ejercicios y ejemplos, podremos instalar lo que se llama un "fake SMTP server" o lo que es lo mismo, un servidor de SMTP falso. También encontrarás este tipo de software con el nombre "dummy SMTP".

Básicamente consiste en un programa que se ejecuta en local, que tiene abiertos los puertos típicos de SMTP para conexiones en local y que responde como un servidor SMTP de verdad. Sin embargo, es falso porque no tiene las funcionalidades de envío del correo. Es decir, los correos realmente no se envían, se quedan en el registro de "procesos de envío" del servidor fake smtp para que podamos revisar los mensajes salientes y podamos analizar su forma, contenido, cabeceras, etc. De este modo, aunque los mensajes no lleguen a sus destinatarios, podemos estar seguros de que la parte de la programación está funcionando correctamente.

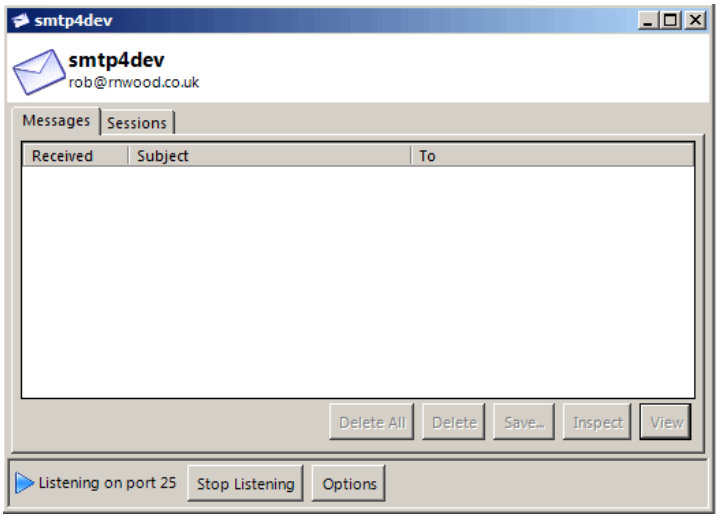
Esta opción se suele usar cuando nuetro proyecto está en modo producción.

Las aplicaciones fake SMTP que podemos encontrar son:

- **smtp4dev**
- **papercut**

10.2.1 smtp4dev

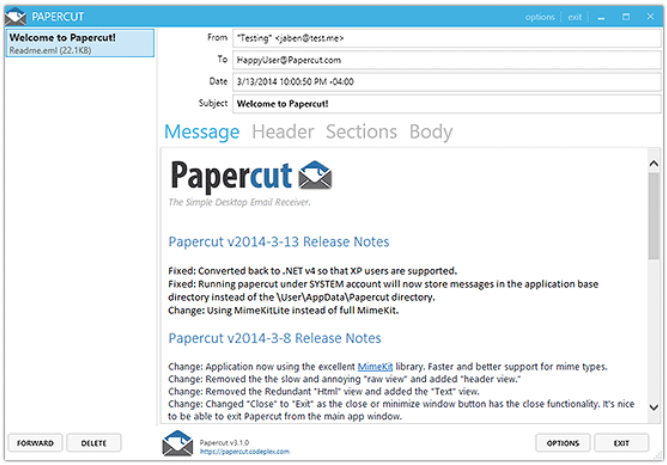
Es un programa para Windows. Lo descargas, lo descomprimes y lo sitúas en cualquier lugar. Realmente no se tiene que instalar, es un sencillo ejecutable .exe que, una vez puesto en marcha, se queda como programa residente.



Lo podrá descargar en la siguiente dirección:

- http://smtp4dev.codeplex.com/

10.2.2 papercut



Proporciona un servidor SMTP local y captura todo el correo electrónico enviado a través de él, en lugar de enviarlo a los destinatarios reales.

Para descarga, instalación y configuración acceda a los siguientes enlaces:

- https://www.papercut.com/

10.3 Servidor SMTP

Si queremos enviar email desde una aplicación PHP en producción no nos queda más remedio que usar un servidor SMTP. Para ello tenemos varias posibilidades entre ellas destacamos las siguientes:

- Servidor SMTP de pago SENDINBLUE.COM
- Servidor SMTP gratuito SMTP.GMAIL

10.3.1 Sendinblue.com

Una vez registrado suministra de forma gratuita un servidor SMTP restringido, es decir, que limita el número de mensajes a enviar y el uso de algunas funciones con PHPMailer. Es bastante profesional y en caso necesario sería una buena opción pagar para adquirir todas las funcionalidades.

10.3.2 Smtp.gmail

Es un servicio SMTP que ofrece gmail de forma totalmente gratuita a todo el que tenga una cuenta en google.

Para poder usarla hay que realizar unos sencillos pasos de configuración en nuestra cuenta de gmail y luego usar adecuadamente los parámetros con la clase PHPMailer que estudiaremos más adelante.

10.4 Función mail()

Para el envío de correos electrónicos utilizando PHP disponemos de una función bastante potente, incluida en todas las versiones de PHP, sin necesidad de instalar ningún añadido. Su comportamiento es bien sencillo, aunque en el trasfondo hay mucha información adicional para asegurarnos que el correo se entrega y para configurar PHP y el servidor SMTP que vayas a usar.

La función se llama **mail()** y recibe tres parámetros de manera obligada y otros dos parámetros que podemos colocar opcionalmente. Devuelve true si se envió el mensaje correctamente y false en caso contrario.

La sintaxis es la siguiente

```
bool mail ( string $to , string $subject , string $message [, string
$additional_headers [,string $additional_parameters ]] )
```

Primero observemos los parámetros de **obligado** uso para la función mail() de PHP, que son los mismos que necesitarías en cualquier tipo de mensaje.

- To o Destinatario. La dirección de correo o direcciones de correo que han de recibir el mensaje. Si incluimos varias direcciones debemos separarlas por una coma.
- Subject o Asunto: para indicar una cadena de caracteres que queremos que sea el asunto del correo electrónico a enviar.
- Message o Cuerpo: el cuerpo del mensaje, lo que queremos que tenga escrito el correo.

Además, la función mail(), nativa de PHP, te ofrece la posibilidad de indicar otros parámetros, de manera **opcional**. Pero, a pesar de ser simples parámetros opcionales, se recomienda el uso de al menos el primero.

- Headers: Cabeceras del correo. Datos como la dirección de respuesta, las posibles direcciones que recibirán copia del mensaje, las direcciones que recibirán copia oculta, si el correo está en formato HTML, etc.

- Additional_parameters: esta opción no suele utilizarse y, además, sólo está disponible a partir de la versión PHP 4.0.5 y desde PHP 4.2.3 está deshabilitado en modo seguro. Puede usarse para pasar parámetros adicionales al programa configurado para enviar el correo, cuando se manda el mail usando la opción de configuración sendmail_path.

Veamos un sencillo **ejemplo** de uso de la función mail().

```
mail($destinatario, $asunto, $mensaje, "From: Remitente
<remitente@tudominio.com>");
```

Para mantener el código ordenado es mejor separar los contenidos de los parámetros en variables, veamos ahora el ejemplo completo.

```
//Destinatario
$destinatario = "destinatario@dominio.com";

//Asunto
$asunto = "La función mail() de PHP";

//Mensaje
$mensaje = "Aquí aprenderán cómo enviar mensajes con la función <b>mail()</b>
de PHP con código HTML incrustado!";

//Cabecera. Debe establecerse la cabecera Content-type
$header = "Mime-Versión: 1.0". "\r\n";
$header .= "Content-Type: text/html; charset-iso-8859-1"."\r\n";

//Cabeceras adicionales
$header .= "To: Juan Carlos <nerom24@gmail.com>\n"; //dirección respuesta
$header .= "From: J. Carlos<nerom24@hotmail.com>\n"; //remitente
$header .= "cc: juanito@gmail.com"."."\r\n";  //Con copia
$header .= "Bcc: maria@hotmail.com"."\r\n";  //Con copia oculta

//
If (mail($destinatario , $asunto , $mensaje , $header ))
       echo('<p>Email enviado.</p>');
} else {
       echo('<p>Error de Envío.</p>');
};
```

Ten en cuenta que las líneas del "header" del correo se separan con "\n". Lo más importante en los correos HTML es el elemento del header "Content-Type: text/html". Comillas (") en el mensaje tienen que ser enmascaradas con Backslashs (\).

El elemento del header "To: …" es opcional y muy útil en caso que la dirección de contacto para clientes es diferente de la dirección que envió un newsletter – por ejemplo.

Veamos otro **ejemplo** sencillo en el que se usa la función mail() a partir del contenido de un formulario

```
<html>
        <head>
        <title>Ejemplo de email en PHP desde Formulario</title>
        </head>

        <body>
        <?php

                if($_POST)
                {
                        // Correo al que queremos que llegue
```

```
                        $destinatario = "TU_CORREO_AQUI@gmail.com";
                        // Asunto
                        $asunto = "Email de prueba del Tutorial PHP 5";
                        // Mensaje
                        $mensaje = "Hola, este email es una prueba del Tutorial PHP 5. Los datos
anexos al email son: <br><br>
                                                Nombre: ".$_POST['nombre']."<br>
                                                Ciudad: ".$_POST['ciudad']."<br>
                                                Año de nacimiento:
".$_POST['anoNacimiento']."<br><br>

                                                Saludos!";
                        // Cabeceras
                        // Para enviar un correo HTML, debe establecerse la cabecera Content-type
                        $cabeceras  = 'MIME-Version: 1.0' . "\r\n";
                        $cabeceras .= 'Content-type: text/html; charset=iso-8859-1' . "\r\n";
                        // Cabeceras adicionales
                        $cabeceras .= 'To: TU NOMBRE <TU_CORREO_AQUI@gmail.com>' . "\r\n";
                        $cabeceras .= 'From: Tutorial PHP 5 <tutorial@tutorialphp.net>' . "\r\n";
                        // Enviamos el email

                        if(@mail($destinatario, $asunto, $mensaje, $cabeceras))
                        {
                                echo "El email se envió correctamente a ".$destinatario.".";
                        }
                        else{
                                echo "El email no se pudo enviar.";
                        }

                }

        ?>

                <form action="" method="post">
                <p>
                Dime cuál es tu nombre: <input type="text" name="nombre">
                </p>
                <p>
                Dime cómo se llama tu ciudad: <input type="text" name="ciudad">
                </p>
                <p>
                ¿En qué año naciste?:
                        <select name="anoNacimiento">
                                <option value="null">Selecciona un año</option>
                                <?php
                                $anos = 1900;
                                        while ($anos < 2010)
                                                {
                                        ?>
                                        <option value="<?=$anos;?>"><?=$anos;?></option>
                                                <?php
                                                        $anos++;
                                                        }
                                                ?>
                        </select>
                </p>
                        <p>
                        <input type="submit" value="Validar">
                </p>

        </form>

        </body>
        </html>
```

10.5 Clase PHPMailer

PHPMailer es una clase creada específicamente para hacer sencillo el envío de emails que tienen características complejas, siendo bastante útil y potente. Permite enviar email con archivos adjuntos, diferentes tipos de servidores SMTP (con o sin autenticación de usuario), a

la vez que da soporte a otras características de la función mail() nativa de PHP, pero de una manera un poco más simple.

Está disponible con licencia de libre distribución y uso, código abierto y se puede obtener a través del enlace siguiente:https://github.com/PHPMailer/PHPMailer.

10.5.1 Instalación

Bajamos el script mediante el archivo PHPMailer-master.zip y descomprimimos el archivo en una carpeta por ejemplo, /phpmailer.

En la cabecera de nuestro procedimientos php incluimos la clase PHPMailer.

```
//Incluimos la clase de PHPMailer
require_once('phpmailer/class.phpmailer.php');
```

10.5.2 Juego Caracteres UTF-8

Para evitar problemas

```
$mail->CharSet = "UTF-8";
$mail->Encoding = "quoted-printable";
```

10.5.3 Ejemplo Básico

```
<? Php
 Requiere 'PHPMailerAutoload.php' ;
 $mail = new PHPMailer;
 $mail->setFrom ( 'from@example.com' , 'Tu nombre' );
 $mail->AddReplyTo('respuesta@example','Email respuesta');
 $mail->addAddress ( 'myfriend@example.net' , 'Mi Amigo' );
 $mail->Subject = 'Primer mensaje de PHPMailer' ;
 $mail->Body = '¡ Hola!  Este es mi primer correo electrónico enviado a través
de PHPMailer.  ' ;
 If ( !$mail->send ()) {
       Echo 'El mensaje no fue enviado.  ' ;
       Echo ' Error de correo: ' .  $ Mail -> ErrorInfo ;
 } Else {
       Echo ' Se ha enviado un mensaje.  ' ;
 }
```

10.5.4 Email en Formato HTML

El envío de correo electrónico HTML es una tarea bastante simple con PHPMailer, aunque puede requerir un conocimiento significativo de HTML. En particular, los clientes de correo varían mucho en su prestación de correo electrónico HTML (mucho más que los navegadores web), con algunos se niegan a mostrarlo por completo. Para aquellos clientes de correo que no pueden mostrar HTML, puede proporcionar un cuerpo de correo electrónico alternativo que contenga el mensaje como texto sin formato.

Primero crearemos un mensaje HTML básico:

```
<? Php
 Requiere ' PHPMailerAutoload.php ' ;
 $Mail = nuevo PHPMailer ;
 $Mail -> setFrom ( ' from@example.com ' , ' Tu nombre ' );
```

```
$Mail -> addAddress ( ' myfriend@example.net ' , ' Mi Amigo ' );
$Mail -> Subject = ' Un mensaje HTML ' ;
$Mail -> isHTML ( true );
$Mail -> Body = ' Hola, <b> mi amigo </ b>!  ¡Este mensaje usa HTML!  ' ;
```

Se llama a $mail->isHTML(true) a continuación podrá usar HTML para $mail->Body .

Para asegurarse de que el destinatario será capaz de leer el correo electrónico, incluso en el caso de que el cliente de correo electrónico no admita HTML, podemos agregar una versión de texto sin formato del mensaje:

```
$Mail -> AltBody = " ¡Hola, amigo! Este mensaje usa texto sin formato! " ;
```

Esto establece el cuerpo AltBody. Si utiliza esta función, el mensaje usará automáticamente el tipo MIME multipart/alternative , de forma que el usuario final podrá elegir formato de lectura.

Tenga en cuenta que la compatibilidad con HTML varía enormemente, pero por lo general sólo puede utilizar HTML puro. Esto significa que no hay secuencias de comandos, no Flash, etc. Lo que puede utilizar está documentado por el proyecto de estándares de correo electrónico.

10.5.5 Archivos Adjuntos

El envío de correos electrónicos de texto sin formato suele ser insuficiente. Tal vez usted necesita adjuntar algo a su correo, como una imagen o un archivo de audio. O quizás necesite adjuntar varios archivos.

Hay tres maneras de adjuntar algo a su correo:

- Adjuntar Archivos
- Adjunto de Cadenas
- Adjunto en Línea

10.5.5.1 Adjuntar Archivos

El comando para adjuntar un archivo local es $mail->addAttachment($path); , Donde $path contiene la ruta al archivo que desea enviar, y se puede colocar en $mail = new PHPMailer; lugar entre $mail = new PHPMailer; Y enviar el mensaje. Tenga en cuenta que no puede utilizar una dirección URL para la ruta : sólo puede utilizar la ruta del sistema de archivos local. Consulte las notas sobre los adjuntos de cadenas a continuación para saber cómo utilizar el contenido remoto.

La ruta $ ser una relativa (de su secuencia de comandos, no la clase PHPMailer) o una ruta completa al archivo que desea adjuntar. Si desea enviar contenido desde una base de datos o una API web (por ejemplo, un generador de PDF remoto), no utilice este método; en su lugar, utilice addStringAttachment .

Si desea más opciones o desea especificar la codificación y el tipo MIME del archivo, puede utilizar otros tres parámetros, todos ellos opcionales:

```
$Mail -> addAttachment ( $path , $name , $coding, $type );
```

- $name es un parámetro opcional, que se utiliza para establecer el nombre del archivo que se incrustará en el correo electrónico. La persona que recibirá su correo sólo verá este nombre, en lugar del nombre del archivo original.
- $encoding Codificación es un poco más técnico, pero con este parámetro se puede establecer el tipo de codificación del archivo adjunto. El valor base64 es base64 . Otros tipos que se admiten incluyen: 7bit 8bit , 8bit , binary y quoted-printable . Generalmente cualquier archivo binario (por ejemplo, una imagen) debe usar base64 ; Los archivos adjuntos basados en texto normalmente usan quoted-printable .
- $type es el tipo MIME del archivo adjunto. Los tipos de contenido se definen no necesariamente por sufijos de archivo (es decir, .jpg o .mp3), pero se utiliza un tipo MIME (MIME = Multipurpose Internet Mail Extensions). Este parámetro hace posible cambiar el tipo MIME de un archivo adjunto del valor application/octet-stream de application/octet-stream (que funciona con todo tipo de archivo, pero significa que el receptor no puede manejarlo correctamente) a un tipo MIME más específico, como image/jpeg para una foto JPEG, por ejemplo. Por lo general, no necesita configurarlo usted mismo, ya que PHPMailer lo asignará automáticamente desde la extensión del archivo.

10.5.5.2 Adjunto de Cadenas

El método addStringAttachment() funciona igual que addAttachment() , pero pasa el addAttachment() real del elemento en lugar de una ruta del sistema de archivos. El parámetro $filename es necesario, ya que se utiliza para proporcionar un nombre de archivo para los datos de cadena en el extremo del receptor.

También acepta los mismos parámetros que se han descrito anteriormente.

Entonces, ¿por qué usar addStringAttachment() en lugar de addAttachment() ? ¿Es para archivos de sólo texto? No, en absoluto. Es principalmente para bases de datos y otros contenidos que no son archivos. Los datos almacenados en una base de datos se almacenan siempre como una cadena (o un objeto BLOB: Binary Large). Puede consultar su base de datos para una imagen almacenada como BLOB y pasar la cadena resultante a addStringAttachment() .

Si desea utilizar una URL remota para obtener su contenido (por ejemplo, obtener contenido PDF desde una URL remota), haga lo siguiente:

```
$Mail -> addStringAttachment ( file_get_contents ( $url ), 'myfile.pdf' );
```

10.5.5.3 Adjunto en Línea

Si desea hacer un correo electrónico HTML que se refiera a imágenes que también se adjuntan al mensaje, es necesario adjuntar la imagen con un identificador de contenido y, a continuación, vincular a la etiqueta a la misma. Por ejemplo, si agrega una imagen como un archivo adjunto en línea con el ID de contenido de my-photo , accedería a ella dentro del cuerpo HTML utilizando <img src="cid:my-photo" alt="my-photo"> .

En detalle, aquí está la función para agregar un archivo adjunto incrustado:

```
$Mail -> addEmbeddedImage ( $filename , $cid );
```

El proceso de conectar etiquetas de imagen a identificadores de contenido es un poco complicado, pero el método msgHTML() puede hacer la mayor parte del trabajo para usted.

Para obtener más información sobre el correo electrónico HTML, consulte la sección Uso del correo electrónico HTML.

10.5.6 Uso servidor SMPT

Si deseas utilizar un servidor SMTP, tenemos que agregar unas cuantas lineas despues de crear la instancia.

```
$correo = new PHPMailer();

//Le decimos al script que utilizaremos SMTP
$correo->IsSMTP();

//Activaremos la autentificación SMTP el cual se utiliza en la mayoría de casos
$correo->SMTPAuth = true;

//Especificamos la seguridad de la conexion, puede ser SSL, TLS o lo dejamos en blanco
si no sabemos
$correo->SMTPSecure = '';

//Especificamos el host del servidor SMTP
$correo->Host = "mail.misitio.com";

//Especficiamos el puerto del servidor SMTP
$correo->Port = 25;

//El usuario del servidor SMTP
$correo->Username   = "usuario@misitio.com";

//Contraseña del usuario
$correo->Password   = "mipassword";
```

A continuación veamos un ejemplo completo de como usar GMail como servidor SMTP.

```
<?php

require_once('phpmailer/class.phpmailer.php');
$correo = new PHPMailer();
$correo->IsSMTP();
$correo->SMTPAuth = true;
$correo->SMTPSecure = 'tls';
$correo->Host = "smtp.gmail.com";
$correo->Port = 465;
$correo->Username   = "micuenta@gmail.com";
$correo->Password   = "mipassword";
$correo->SetFrom("micuenta@gmail.com", "Mi Codigo PHP");
$correo->AddReplyTo("micuenta@gmail.com","Mi Codigo PHP");
$correo->AddAddress("destino@correo.com", "Jorge");
$correo->Subject = "Mi primero correo con PHPMailer";
$correo->MsgHTML("Mi Mensaje en <strong>HTML</strong>");
$correo->AddAttachment("images/phpmailer.gif");

if(!$correo->Send()) {
  echo "Hubo un error: " . $correo->ErrorInfo;
} else {
```

```
  echo "Mensaje enviado con exito.";
}

?>
```