

Módulo Tercero
*Conceptos avanzados de
PHP*

- PHP Orientado a Objetos
 - Definiciones:
 - **clase**: conjunto de propiedades(variables) y métodos (funciones) agrupadas en 1 entidad instanciable. (definición general de un tipo de objeto).
 - **objeto**: instancia de una clase en tiempo de ejecución.
 - **herencia**: cualidad de una clase mediante la cual una clase hereda (puede usar) propiedades y métodos de la clase padre.
 - **constructor**: método que se ejecuta al instanciar un objeto. (su nombre coincide con el nombre de una clase).

- Ejemplo definición de una clase:

```
<?php
class formulario {
    var $name; // Propiedades del objeto.
    var $type = "text";
    function formulario() {
        //Método constructor vacio
    }
    function setName($name) {
        $this->name = $name;
        // Accedo a una propiedad de la clase
        // la variable $this se refiere al objeto.
    }
    function dibujar() {
        echo "<input type=\"".$this->type."\"
name=\"".$this->name."\" />";
    }
} ?>
```

Técnicas Avanzadas Web

Módulo 3: PHP

- Ejemplo instanciar clase:

```
<?php
```

```
$f1 = new formulario();
```

```
$f2 = new formulario();
```

```
$f1->setName("nombre");
```

```
$f2->setName("Telefono");
```

```
$f1->dibujar();
```

```
// dibuja: <input type="text" name="nombre" />
```

```
$f2->dibujar();
```

```
// dibuja: <input type="text" name="telefono" />
```

```
?>
```

- Ejemplo Herencia:

```
class checkbox extends formulario {  
    function checkbox(){  
        $this->type="checkbox";  
    }  
}  
  
$f3 = new checkbox();  
$f3->setName("fumador");  
  
$f3->dibujar();  
// <input type="checkbox" name="fumador" />
```

- Funciones PHP para el manejo de Objetos

- bool **class_exists(str);**

```
if (class_exists("checkbox"))  
    echo "Puedo instanciar un objeto checkbox";
```

- array **get_class_methods(str);**

```
$aMetodos = get_class_methods("checkbox");  
echo "La clase checkbox tiene  
".sizeof($aMetodos)." métodos:<br />";
```

```
foreach($aMethods as $metodo)  
    echo $metodo."<br />";
```

- array **get_class_vars(str);**

```
$aVars = get_class_vars("checkbox");  
echo "La clase checkbox tiene ".sizeof($aVars)." propiedades:<br />";  
foreach($aVarsas $var)  
    echo $var."<br />";
```

– **str get_class(obj);**

```
$f1 = new formulario();  
echo get_class($f1); // formulario;
```

– **str get_parent_class(obj|str);**

```
$f2 = new checkbox();  
echo get_parent_class($f2); // formulario  
echo get_parent_class("checkbox"); // formulario
```

– **bool is_a(obj,str);**

```
$f2 = new formulario();  
if (is_a($f2,"checkbox")) echo "f2 no es un objeto  
instanciado de checkbox";  
else echo "f2 es n objeto instancia de checkbox";
```

- Tratamiento de imágenes (librería GD)
 - PHP no solo se utiliza para **generar** texto plano (HTML), sino que también puede generar datos binarios (imágenes, pdf, swf, etc).
 - La principal diferencia frente al texto plano convencional, es que debemos especificar la cabecera de respuesta HTTP “**Content-type**”, para sobrescribir el mime-type por defecto (text/html), por el mime-type del archivo que queremos mostrar.

```
header("Content-type: image/png");  
header("Content-type: image/jpg");
```

- **getimagesize(\$fichero)**
 - Obtiene información de un fichero tipo imagen, y nos la devuelve en un array.
 - Devuelve **false** si el fichero no se trata de una imagen (muy útil para filtrar subidas de imagenes).

```
<?php
```

```
    $infoFoto = getimagesize("imagen.jpg");  
    echo $infoFoto[0]; // anchura en pixels  
    echo $infoFoto[1]; // altura en pixels  
    echo $infoFoto[2]; // tipo de imagen  
  
    1 = GIF, 2 = JPG, 3 = PNG, 4 = SWF, 5 = PSD,  
    6 = BMP, 7 = TIFF(intel), 8 = TIFF(motorola),  
    9 = JPC, 10 = JP2, 11 = JPX, 12 = JB2,  
    13 =SWC, 14 = IFF, 15 = WBMP, 16 = XBM  
    echo $infoFoto['mime']; //devuelve el mime-  
    type de la imagen
```

```
?>
```

- **imagecreatetruecolor(x,y)**
 - Crea una imagen en memoria de anchura “x” y altura “y”; devuelve el **identificador** de dicha imagen.
- **imagecreatefrompng(\$ruta)**
 - Crea una imagen a partir de un fichero **png**, devolviendo su identificador.
- **imagecreatefromjpeg(\$ruta)**
 - Crea una imagen a partir de un fichero **jpeg**, devolviendo su identificador.
- **imagecreatefromgif(\$ruta)**
 - Crea una imagen a partir de un fichero **gif**, devolviendo su identificador.

- **imagepng**(\$imagen,[\$ruta])
 - Devuelve una imagen (su contenido binario) en formato png bien por la salida estándar (navegador) o si está incluido, la escribe en la ruta especificada como segundo parámetro.
- **imagejpeg**(\$image,[\$ruta],[calidad])
 - Devuelve la imagen en formato jpeg, pudiendo incluir además del fichero, la calidad (0-100) de ficho jpeg.
- **imagegif**(\$image,\$ruta)
 - Devuelve la imagen en formato gif. (No soporta gif animado).

Técnicas Avanzadas Web

Módulo 3: PHP

- **imagesx(\$im)**
 - Devuelve la anchura en pixels de un identificador de imagen.
- **imagesy(\$im)**
 - Devuelve la anchura en pixels de un identificador de imagen.
- **imagecolorallocate(\$im,\$r,\$g,\$b);**
 - Vincula un color expresado en sus componentes RGB, con un identificador de imagen. (para poder usarlo)
- **imagecolorallocatealpha(\$im,\$r,\$g,\$b,\$alpha);**
 - Igual que imagecolorallocate, pero además con soporte de alpha(transparencia) expresado de 0 a 127 (0 = sin transparencia, 127= totalmente transparente).

- **imagerectangle**(\$im,\$x1,\$y1,\$x2,\$x3,\$color)
 - Dibuja un rectángulo en la imagen, en las coordenadas y del color especificados.
 - La esquina superior izquierda estará en el punto \$x1,\$y1 y la esquina inferior derecha en el punto \$x2,\$y2.
- **imageline**(\$im,\$x1,\$y1,\$x2,\$y2,\$color)
 - Dibuja una línea desde el punto \$x1,\$y1 hasta el punto \$x2,\$y2.
- **imageellipse**(\$im,\$cx,\$cy,\$ancho,\$alto,\$color)
 - Dibuja una eclipse con centro en el punto \$cx,\$cy y ancho y alto especificados en \$ancho y \$alto.

Técnicas Avanzadas Web

Módulo 3: PHP

- **imagepolygon(\$im,\$aPuntos,\$numPuntos,\$c)**
 - Dibuja un polígono uniendo secuencialmente el array de puntos.

```
$aPuntos= array(  
    1,  1,  // punto 1 (x, y)  
    10, 1,  // punto 2 (x, y)  
    1,  10 // punto 3 (x, y)  
);  
imagepolygon($im, $aPuntos, 3, $color);
```

- **imagefilledrectangle**
- **imagefilledellipse**
- **imagefilledpolygon**
 - Actuarán igual que imagerectangle, imageellipse e imagepolygon, con la diferencia que rellenarán también el área que queda dentro de las formas dibujadas.

- **imagestyle(\$im,\$aColores)**
 - Establece el estilo de las líneas con las que se dibujarán las formas geométricas.
 - En la función de dibujo geométrico deberá especificarse la constante **IMG_COLOR_STYLED** en lugar del color.

```
// siendo $r el color rojo asociado a imagen y
// $b el color blanco...
$estilo= array($r,$r,$r,$b,$b,$b);
imagestyle($im, $estilo);
imageline($im, 0, 0, 0, 10, IMG_COLOR_STYLED);
// Dibujará una línea compuesta de 3 pixels rojos
y 3 pixels blancos.
```

- **imagecopyresized**(\$imDestino,\$imOrigen,\$xDestino,\$yDestino,\$xOrigen,\$yOrigen,\$anchoDestino,\$altoDestino, \$anchoOrigen,\$altoOrigen)
 - Copia resizeando una porción de una imagen origen (desde un punto determinado y unas medidas determinadas) a una porción exacta de una imagen destino.
- **imagecopyresampled**(\$imDestino,\$imOrigen,\$xDestino,\$yDestino,\$xOrigen,\$yOrigen,\$anchoDestino,\$altoDestino, \$anchoOrigen,\$altoOrigen)
 - Copia **resampleado** una porción de una imagen origen (desde un punto determinado y unas medidas determinadas) a una porción exacta de una imagen destino.
 - Se **recomienda** utilizar esta ya que suaviza los valores de los pixeles mediante interpolación, manteniendo buena claridad al resizear una imagen.

- **imagefttext**(\$im,\$tam,\$angulo,\$x,\$y,\$color,\$fuente,\$texto)
 - Escribe el valor de la cadena \$texto (UTF8), en la imagen \$im.
 - \$tam especifica el tamaño en puntos de la fuente utilizada
 - \$x e \$y especifican el punto en el que se comienza a escribir la cadena. El punto coincide con esquina inferior izquierda del texto, no de la parte más baja del texto.
 - \$color será el color que se utilice para escribir el texto.
 - \$fuente será la ruta al archivo ttf que utilicemos.

- **imagebbox(\$im,\$tam,\$angulo,\$x,\$y,\$color,\$fuente,\$texto)**
 - Devuelve un array con las posiciones de la caja de texto. No dibuja nada en la imagen.
 - 0 esquina inferior izquierda, posición X
 - 1 esquina inferior izquierda, posición Y
 - 2 esquina inferior derecha, posición X
 - 3 esquina inferior derecha, posición Y
 - 4 esquina superior derecha, posición X
 - 5 esquina superior derecha, posición Y
 - 6 esquina superior izquierda, posición X
 - 7 esquina superior izquierda, posición Y

- **Utilización de img2Thumbnail**
 - **Clase que crea un thumbnail a partir de una ruta a una imagen.**

<http://www.phpclasses.org/browse/package/1041.html>

- **Recibe como parámetros obligatorios la imagen origen y el tamaño de la imagen destino. Como parámetros adicionales se puede indicar una variable para que ajuste el radio inicial modificando el tamaño destino, o añadiendo un fondo a la imagen, cuyo color debe ser pasado como parámetro.**

```
require("class.img2thumb.php");  
$i = new Img2Thumb  
($ruta, 200, 200, "", true, 255, 255, 255);
```

- Creación dinámica de PDF
 - Aunque existen funciones nativas de php para la generación de ficheros pdf [1], la API que nos ofrece PHP es muy poco usable e incómoda.
 - Existe una clase libre (freeware), que no requiere ninguna configuración o módulo especial y que cumple con creces las necesidades: fpdf [2]
 - En el sitio de la página, existe documentación[3], ejemplos y extensiones [4] disponibles para manejar esta clase.

[1] <http://es2.php.net/manual/es/ref.pdf.php>

[2] <http://www.fpdf.org>

[3] <http://fpdf.org/es/doc/index.php>

[4] <http://fpdf.org/en/script/index.php>

- Ejemplo fpdf:

```
<?php
```

```
require ( 'fpdf153/fpdf.php' );
```

```
$pdf=new FPDF();
```

```
$pdf->AddPage();
```

```
$pdf->SetFont('Arial','B',16);
```

```
$pdf->Cell(110,10,'Hello World!');
```

```
$pdf->Image('imagen.jpg',10,20,33);
```

```
$pdf->Output();
```

```
?>
```

- **Creación de PDFs partiendo de una plantilla (FPDI)**
 - Se trata de una clase extendida de fpdf, que acepta la importación de pdfs en forma de plantillas.
 - Licencia Libre (Licencia Apache v2.0)
 - Al tratarse de una clase hija de fpdf, cualquier objeto instanciado a partir de esta clase, poseerá todos los métodos de fpdf.

[1] <http://fpdi.setasign.de/index.php>

- **Ejemplo FPDF**

```
<?php
```

```
ini_set("include_path", " ../fpdf153:../fpdi1.1");  
require('fpdi.php');
```

```
$pdf= new fpdi();  
$pagecount = $pdf->setSourceFile("frikada.pdf");  
$tplidx = $pdf->ImportPage(1);  
$pdf->addPage();  
$pdf->useTemplate($tplidx);  
$pdf->Image('bush.jpg',10,20,33);  
$pdf->Output("newpdf.pdf","I");  
$pdf->closeParsers();
```

```
?>
```

- **PEAR**

- PHP Extension and Application Repository
- Librería estructurada PHP de código abierto, mantenido por la comunidad.
- Sistema standard de mantenimiento y actualización de paquetes.
- Estilo de código específico.
- Los paquetes, aunque son independientes, pueden tener dependencias unos de otros; el instalador se encarga de satisfacerlas.

- **Instalación de PEAR (windows)**

- Hay que asegurarse que la versión del PHP que tenemos instalada, ha sido descargada con soporte para PEAR. En el directorio “php”, debe existir otro directorio llamado “PEAR”.
- Para instalar PEAR, habrá que ejecutar el fichero .bat “go-pear.bat”, o simplemente desde un prompt el siguiente comando:

```
c:\php\php.exe c:\php\PEAR\go-pear.php
```

- Aunque el instalador es bastante sencillo, habrá que asegurarse de especificar bien las rutas.
- Una vez concluida la instalación, deberemos ejecutar fichero “PEAR_ENV.reg”, para añadir al registro de windows, ciertas variables de entorno.
- PEAR viene instalado por defecto con los siguientes paquetes: DB, Net_Socket, Net_SMTP, Mail, XML_Parser, PHPUnit.

- Primeros Pasos con PEAR

- Si hacemos `phpinfo()`; en un script de prueba, veremos que la directiva `include_path`, deberá incluir el directorio a nuestro repositorio de PEAR:

```
include_path = include_path  
;C:\AppServ\php\PEAR\pear
```

- Deberemos tenerlo en cuenta si modificamos esa directiva, y a su vez utilizamos una clase de PEAR:

```
ini_set("include_path", ini_get("include_path").":c  
:\NuevaRuta\");
```

- Instalando el primer paquete PEAR:
SpreadSheet_Excel_writer
 - Desde el directorio c:\rutaPHP\PEAR, ejecutamos:

```
C:\AppServ\php\PEAR>pear search excel
```

```
MATCHED PACKAGES:
```

```
=====
```

```
PACKAGE                                STABLE/(LATEST)    LOCAL
Spreadsheet_Excel_Writer 0.9.0/(0.9.0beta)
Package for generating Excel spreadsheets
```

- Al intentar instalarlo nos ocurrirá lo siguiente:

```
C:\AppServ\php\PEAR>pear install
Spreadsheet_Excel_Writer
```

```
No release with state equal to: 'stable' found for
'Spreadsheet_Excel_Writer'
```

- Lo que significa que el paquete no está en su versión estable. Aún así, queremos instalarlo.

Técnicas Avanzadas Web

Módulo 3: PHP

- Forzamos a Instalar, añadiendo el número de versión:

```
C:\AppServ\php\PEAR>pear install
Spreadsheet_Excel_Writer-0.9.0
downloading Spreadsheet_Excel_Writer-0.9.0.tgz ...
Starting to download Spreadsheet_Excel_Writer-
0.9.0.tgz (55,541 bytes)
.....done: 55,541 bytes
requires package `OLE' >= 0.5
Spreadsheet_Excel_Writer: Dependencies failed
```

- Al estar forzando, no instala las dependencias:

```
C:\AppServ\php\PEAR>pear install OLE-0.5
downloading OLE-0.5.tgz ...
Starting to download OLE-0.5.tgz (9,058 bytes)
.....done: 9,058 bytes
install ok: OLE 0.5
```

Técnicas Avanzadas Web

Módulo 3: PHP

- Finalmente instalamos la clase Spreadsheet_Excel_Writer:

```
C:\AppServ\php\PEAR>pear install
Spreadsheet_Excel_Writer-0.9.0
downloading Spreadsheet_Excel_Writer-0.9.0.tgz ...
Starting to download Spreadsheet_Excel_Writer-
0.9.0.tgz (55,541 bytes)
.....done: 55,541 bytes
install ok: Spreadsheet_Excel_Writer 0.9.0
```

- Creando nuestra primera hoja de excel:o

```
<?php
```

```
require_once('Spreadsheet/Excel/Writer.php');
```

```
// Instanciamos la clase Spreadsheet_Excel_writer  
en el objeto
```

```
$MiExcel = new Spreadsheet_Excel_Writer();
```

```
// El método send enviará las cabeceras HTTP de  
respuesta necesarias al cliente, indicándole ese  
nombre de fichero.
```

```
$MiExcel->send('test.xls');
```

Técnicas Avanzadas Web

Módulo 3: PHP

```
// Creación de Hojas distintas dentro del mismo
Excel. Cada Hoja será otro objeto de tipo
Worksheet. Este objeto es instanciado mediante un
método del objeto, por ello es asignado por
referencia (&), para que sea exactamente el
objeto que devuelve el método addWorksheet, y no
una copia (como pasa con las asignaciones)
```

```
$hoja1 =& $MiExcel->addWorksheet('Mi primera
Hoja');
```

```
// El método write recibe como argumentos:
// fila / columna / Texto
$hoja1->write(0, 0, 'DNI');
$hoja1->write(0, 1, 'Nombre');
$hoja1->write(1, 0, '12345678');
$hoja1->write(1, 1, 'Pepito de los palotes');
$hoja1->write(2, 0, '87654321');
$hoja1->write(2, 1, 'Pepito grillo');
```

```
// y por último cerramos, y se enviará la
//navegador
$miExcel->close();
?>
```

- Si quisieramos, salvar el Excel en un fichero, en lugar de mandarlo al navegador, deberíamos no evocar el método send, y especificar en el constructor, la ruta destino del archivo excel:

```
$MiExcel = new Spreadsheet_Excel_Writer($ruta);
//$MiExcel->send('test.xls');
```

- Para añadir formato a las celdas (color, negrita etc...):

```
$fomato1 = & $miExcel->addFormat(array(
                                'Size' => 10,
                                'Align' => 'center',
                                'Color' => 'white');
$formato2 = &$miExcel->addFormat();
$formato2->setBold();
$Hojal->write(4,1,"Texto ejemplo",$formato1);
```

- Más información acerca de otros métodos para Spreadsheet_Excel_Writer en
 - <http://pear.php.net>

- PECL
 - PHP Extension Community Library
 - Repositorio de Extensiones para el core de PHP.
 - Son extensiones en C, que se instalan como módulos de PHP aportando nuevas funcionalidades.
 - El soporte para la librería GD o para MySQL es una extensión al núcleo de PHP.
 - Proyecto hermanoado con PEAR; formó parte de PEAR hasta que se separó (Octubre 2003).

- Funcionamiento:

- Una extensión compilada esta lista para ser usada, simplemente añadiendola en el php.ini:

```
extension = extension.dll // windows
```

```
extension = extension.so // Entornos Unix
```

- También es posible cargarla en tiempo de ejecución con la instrucción dl:

```
if (!extension_loaded('sqlite')) {  
    if (strtoupper(substr(PHP_OS, 0, 3)) === 'WIN') {  
        dl('php_sqlite.dll')  
    } else {  
        dl('sqlite.so');  
    }  
}
```

– Instalación:

- En entornos *nix, existe una herramienta que permite compilar e instalar cualquier extensión de manera ágil y sencilla:

```
tar xvzf ext-xxx.tgz
cd ext-xxx
phpize
./configure && make && make install
```

- Con estos simples comandos, ya tendríamos creada la extensión (.so), y la tendríamos lista para funcionar.

Técnicas Avanzadas Web

Módulo 3: PHP

- Para windows, es necesario encontrar la dll específica para nuestra plataforma.
- Si bien, en la mayoría de paquetes PHP para windows vienen incluidas un gran número de extensiones, podemos descargarlas de la siguiente dirección:
 - <http://pecl4win.php.net>
- Una vez descargado el fichero dll, debemso dejarlo en el directorio **extensions** de nuestro directorio **php** raiz.

- Ejemplo de uso: `php_zip.dll`
 - Librería que permite leer y extraer los ficheros dentro de un fichero comprimido zip.
 - Primero deberemos comprobar que la extensión existe en nuestro directorio **extensions**.
 - Mediante **phpinfo()**, debemos encontrar:
 - Zip support: enabled
 - Funciones proporcionadas por `php_zip.dll`:
 - `zip_open($ruta);`
 - Devuelve false si el fichero no es un fichero zip válido, o un identificador de recurso zip si el fichero es válido.
 - `zip_read($recurso_zip);`
 - Devuelve una variable tipo recurso, que apunta al siguiente fichero en el zip; o false si no hay más ficheros para descomprimir;

- `zip_entry_name($recursoFicheroZip);`
 - Devuelve el nombre del fichero comprimido.
 - Si el fichero se encuentra dentro de un directorio en el propio fichero zip, esta función nos devolverá la ruta relativa, incluyendo el/los nombres de los directorio, seguido al final del nombre de fichero.
 - **basename**(\$nFicheroZip), nos devolverá el nombre del fichero comprimido.
 - **dirname**(\$nFicheroZip), nos devolverá el/los directorios dentro de los cuales está el fichero en cuestión.
- `zip_entry_filesize($recursoFicheroZip);`
 - devuelve el tamaño sin comprimir del fichero.
- `zip_entry_compressedsize($recursoFicheroZip);`
 - devuelve el tamaño comprimido del fichero.

Técnicas Avanzadas Web

Módulo 3: PHP

- `zip_entry_open($recursoZip,$recursoFichero,"r")`
 - Abre el fichero dentro del zip para lectura
- `zip_entry_read($recursoFichero,$bytes)`
 - Devuelve el número de bytes especificado como 2º parámetro, del fichero comprimido.
 - En \$bytes, pasar `zip_entry_filesize($recursoFichero)`, para obtener todo el fichero.
- `zip_entry_close($recursoFichero)`
 - Cierra el fichero Comprimido abierto para lectura, abierto mediante `zip_entry_open($zip,$ficheroZip)`.
- `zip_close($recursoZip)`
 - Cierra el apuntador al fichero zip abierto mediante `zip_open($ruta)`;

- Mail
 - mail(\$destinatario,\$asunto,\$cuerpo,[\$cabeceras],[\$parametros]) es la función php que se encarga de enviar mail a través del smtp configurado.
 - En servidor *nix, este comando buscará el binario de sendmail (o wrapper correspondiente) para enviar el correo.
 - sendmail_path (php.ini)
 - ruta al binario de sendmail. Por defecto la instalación encontrará la ruta adecuada, pero se da la opción a cambiarla.
 - Si esta directiva esta desactivada, PHP buscará el binario sendmail en estas rutas:
/usr/bin:/usr/sbin:/usr/etc:/etc:/usr/ucblib:/usr/lib

- En servidores Windows, la función mail se conecta a través de un socket a un servidor SMTP (exactamente como hace un cliente de correo).
 - **SMTP** (php.ini)
 - servidor SMTP al que php se va a conectar.
 - **smtp_port** (php.ini)
 - puerto del servidor SMTP; 25 por defecto.
 - **smtp_from** (php.ini)
 - Campo From que se utilizará en los mails enviados.
- Este servidor debe tener open-rely (no necesita autenticación).
- Lo más “seguro”, sería instalar un servidor de correo local ligero y escuchando sólo en localhost: QK smtp (<http://qksoft.com>)

Técnicas Avanzadas Web

Módulo 3: PHP

- Utilización de Mail (PEAR)

- En consola, en la ruta adecuada:

```
pear install mail
```

- **factory**(\$tipo_servidor,\$parametros)

- Este método suele ser llamado utilizando el operador de resolución de contexto (::), que se utiliza para acceder a un método de una clase sin tener que instanciarla.
- Devolverá un objeto de la clase Mail, listo para ser enviado:
- Tipos de servidor:

"mail" => se utilizará la función mail de PHP, siendo el array \$parámetros usado como 5º argumento de dicha función.

"sendmail"=> se utilizará el binario de sendmail, especificado en el array de \$parametros:

\$parametros["sendmail_path"]: ruta a sendmail.

\$params["sendmail_args"]: argumentos que se pasarán al hacer la llamada.

Técnicas Avanzadas Web

Módulo 3: PHP

"smtp" => La clase realizará una conexión a un servidor smtp especificado:

```
$params["host"]: server SMTP -localhost
```

```
$params["port"]: puerto SMTP -25
```

```
$params["auth"]: boolean que indica si el server SMTP requiere autenticación.
```

```
$params["username"]: usuario
```

```
$params["password"]: contraseña
```

```
$params["persist"]: boolean que indica si el servidor debe mantener abierta la conexión o no.
```

- `send($destinatarios,$cabeceras,$cuerpo)`
 - Método que enviará el mensaje utilizando el método escogido en la creación del objeto.

Técnicas Avanzadas Web

Módulo 3: PHP

- Ejemplo de uso:

```
include('Mail.php');
$parametros = array(
    "host"=>"localhost",
    "auth"=>true,
    "username"=>"congrio",
    "password"=>"volador");

$destino['to'] = "jabi@irontec.com";

$headers['From'] = "el congrio volador";
$headers['Subject'] = 'Prueba';
$body = 'Cuerpo del mensaje';

$mail_object =& Mail::factory('smtp',
    $parametros);

$mail_object->send($destino, $headers, $body);
```

- Mails Mime-type (html, imágenes,...)
 - El mail fue creado para enviar solamente texto ASCII.
 - Las necesidades de la gente hicieron desarrollar un sistema para poder incluir ficheros binarios en los correos: Correo MIME.
 - Se utiliza el encoding base64, que pasa a texto ASCII cualquier dato binario. Lo cual provoca alrededor de un 33% más de tamaño.
 - A efectos prácticos, un mail MIME, es un mail de texto ASCII, con cabeceras determinadas y un cuerpo construido de manera especial. Pero siempre dependera de un cliente de correo para una correcta interpretación.

- Utilización de Mime_mail (PEAR)
 - En consulta, en la ruta apropiada:

```
pear install Mail_Mime
```
 - Esta clase nos facilitará las cosas a la hora de crear un mensaje en formato HTML y/o con archivos adjuntos. Al final nos devolverá unas cabeceras y un cuerpo concretos que utilizaremos para enviar el mail con el método que escogamos.
 - **Mail_Mime(\$salto_linea)**
 - método constructor de la clase que recibe como parámetro el tipo de salto de línea. Por defecto sera “\r\n”, salto de línea predeterminado en sistemas Windows. Si utilizamos esta clase en sistemas *nix y enviamos el email con la función mail, deberemos especificar “\n” como salto de línea.

- **setTxtBody(\$texto)**
 - Pasa como argumento el texto plano al email. Este texto se mostrará en clientes de correo que no soporten emails en HTML (casi nunca).
- **setHTMLBody(\$html,\$es_fichero)**
 - Se pasará como primer parámetro, bien un fichero o bien un string con todo el HTML del mensaje. Si el segundo parámetro es true, habremos de pasar un fichero y viceversa.
- **addHTMLImage(\$imagen,\$tipo_miem,\$nombre,\$es_fichero)**
 - Si el argumento es_fichero es true, \$imagen deberá ser la ruta a un fichero imagen. \$nombre no se utilizará.
 - Si el argumento de es_fichero es false, \$imagen serán los datos binarios de la imagen y \$nombre será el nombre de la imagen con el que ha sido referenciada desde el HTML.

- **addAttachment**

(\$fichero,\$tipo_mime,\$nombre,\$es_fichero,\$encoding)

- Si \$es_fichero es true, \$fichero deberá ser la ruta al fichero que queremos incluir como adjunto.
- Si \$es_fichero es false, \$fichero serán los datos binarios del fichero, y el nombre de fichero se especificará en \$nombre.
- \$tipo_mime, contendrá el mime_type de dicho fichero (Application/octet-stream por defecto)
- \$encoding: tipo de encoding utilizado para incluir el fichero. Para ficheros binarios suele encodearse en base64 (por defecto), aunque para ficheros de tipo texto se recomienda utilizar encoding “quoted-printable”.

- **get(\$parametros)**
 - Método que devuelve el cuerpo del mensaje construido.
 - El array \$parámetros que recibe es opcional y puede contener un array asociativo especificando diversas opciones de en el mensae:

`$param["text_encoding"]`: encoding para la parte de texto del mensaje (7bit por defecto).

`$param["html_encoding"]`: encoding para el HTML ("quoted-printable" por defecto)

`$param["head_charset"]`: charset para la cabecera (iso-8859-1 por defecto).

`$param["text_charset"]`: charset para el texto del mensaje (iso-8859-1 por defecto).

`$param["html_charset"]`: charset para el HTML (iso-8859-1 por defecto).

- **headers(\$cabecera)**
 - Devuelve la cabecera del mail necesaria para su correcto envío.
 - Recibe como parámetro cabeceras adicionales en un array donde los índices serán los nombre de la cabecera, y el valor de cada índice, el valor de dicha cabecera.

- Ejemplo de uso:

```
include('Mail.php');
include('Mail/mime.php');
$texto = 'Su cliente de correo no soporta mails en
HTML. Actualicese';
$html = '<html><body>Hola que tal?</body></html>';
$eol = "\n";
$headers = array(
    'From'      => 'jabi@jabi.ath.cx',
    'Subject' => 'Hola soy un mensaje de prueba'
);
$mime = new Mail_mime($eol);
$mime->setTXTBody($texto);
$mime->setHTMLBody($html);
$mime->addAttachment("unPDF.pdf,
'application/pdf');
```

Técnicas Avanzadas Web

Módulo 3: PHP

```
$cuerpo = $mime->get();
$cabecera = $mime->headers($hdrs);

$parametros = array(
    "host"=>"localhost",
    "auth"=>true,
    "username"=>"congrío",
    "password"=>"volador");

$destino['to'] = "jabi@irontec.com";

$mail_object =& Mail::factory('smtp',
    $parametros);

$mail_object->send($destino, $cabeceras, $cuerpo);
?>
```

Técnicas Avanzadas Web

Módulo 3: PHP

- Enlaces interesantes
 - <http://www.google.com>
 - <http://www.php.net>
 - <http://pear.php.net>
 - <http://pecl.php.net>
 - <http://www.phpclasses.org>
 - <http://www.phpguru.org>
 - http://www.irontec.com/~jabi/mod3_php.pdf