Redirecciones e inclusiones

PHP también dispone de instrucciones para permitir incluir el contenido de un documento en otro como parte de la respuesta a un cliente, o redirigir a otra página si es necesario.

1. Redirigir a otra página

Las redirecciones se producen cuando desde el cliente accedemos a una página que está preparada para, según ciertas condiciones, enviarnos a otra. Por ejemplo, si intentamos acceder a una zona restringida y no estamos logueados en el servidor, lo normal es que se nos redirija a la página de login para que entremos con nuestro usuario y contraseña.

La redirección de una página a otra se hace aprovechando el protocolo HTTP, mediante una de sus cabeceras, llamada *Location*. Usaremos el comando `header` para acceder a las cabeceras HTTP, y dentro pondremos la cabecera *Location*, junto con la página a la que queremos redirigir. Por ejemplo, si queremos redirigir a la página *login.php* de nuestra web, pondríamos algo como:

```
header("Location:login.php");
```

> **NOTA**: es importante recalcar que las redirecciones deben hacerse *antes* de escribir cualquier contenido por el buffer de salida, es decir, antes de generar cualquier tipo de contenido HTML que enviar al navegador.

También podemos añadir algo más de información con la redirección. Por ejemplo, redirigir pasados X segundos, para así mostrar algún mensaje de advertencia en la página, y que el usuario sepa que va a ser redirigido. Para ello se utiliza la cabecera *Refresh*, indicando el tiempo en segundos a esperar, y un atributo *url* con la página a la que redirigir (si no se pone esto, se recarga la misma página pasado ese tiempo, lo que puede ser útil, por ejemplo, para actualizar resultados deportivos). El siguiente código redirige a *index.php* pasados 5 segundos, mostrando un mensaje de redirección.

```
header("Refresh:5; url=index.php");
...
echo '<p>En breve le redirigiremos a la página principal.</p>';
```

Una de las utilidades prácticas que tiene esta característica es la de poder enviar al usuario a otro recurso si no se cumplen ciertas condiciones. Por ejemplo, si los datos de un formulario no son correctos. Así, podríamos utilizar funciones como `isset` o `empty` (entre otras, como la comprobación de expresiones

regulares) para determinar si los datos recibidos son correctos y, en caso contrario, redirigir a una página de error o al propio formulario de nuevo:

```
if (!isset($_REQUEST['login']) || empty($_REQUEST['login']))
{
    header("Location:login.php");
}
...
```

Ejercicio 1:

Crea una carpeta llamada **ejercicios5** donde ubicar los ejercicios de esta sesión. Copia en ella la carpeta *libros* de la sesión anterior, y modifica la página *result_libros.php* para que, si el texto a buscar que le llega del formulario está vacío, o no le han llegado datos del formulario (por ejemplo, porque hemos cargado esta página directamente, sin pasar por el formulario), redirija a la página *form_libros.php* para rellenar los datos.

Opcionalmente, trata de mostrar un mensaje de error en color rojo, en el formulario, indicando que "Debes rellenar todos los campos antes de enviar el formulario".

1.1. Más sobre las redirecciones

A la hora de trabajar con redirecciones, conviene tener en cuenta algunas cosas. Para empezar, no conviene abusar de ellas, ya que los buscadores suelen castigar este hecho si se hace con frecuencia, penalizando el posicionamiento de la web. Además, reduce el rendimiento al estar haciendo nuevas peticiones para cargar otra página

Además, es recomendable también hacer una llamada a `die()` o `exit()` tras una redirección, para evitar que se siga ejecutando el resto de la página. Por ejemplo, en el siguiente código, aunque fallen todas las comprobaciones, sólo se va a ejecutar la última redirección, porque se sigue interpretando el código de la página y PHP se queda con la última redirección que ha encontrado:

```
if (....)
    header("Location:pag1.php");

if (....)
    header("Location:pag2.php");

if (....)
    header("Location:pag3.php");
```

Para evitar esto, podemos llamar a `die()` justo después de cada redirección. Así, al encontrar una redirección ya deja de ejecutarse el resto de la página:

```
if (....)
{
    header("Location:pag1.php");
    die();
}

if (....)
{
    header("Location:pag2.php");
    die();
}

if (....)
{
    header("Location:pag3.php");
    die();
}
```

2. Incluir páginas dentro de otras

La inclusión de contenido externo es una funcionalidad importante para, por ejemplo y sobre todo, no tener que estar repitiendo código en diferentes páginas. Así, podríamos tener una página *cabecera.php* con el encabezado de nuestra página web (bloque head, logo de la empresa y menú de navegación principal):

```
<!DOCTYPE html>
<html lang="es">
    <head>
        ...
    </head>
    <body>
        <img class="logo" src="imgs/logoempresa.jpg" />
        <ul class="menuNavegacion">
            <li><a href="inicio.php">Inicio</a></li>
            <li><a href="noticias.php">Noticias</a></li>
            ...
        </ul>
```

Después, podríamos incluir este fichero *cabecera.php* en cada página de nuestra web, para así tener la misma cabecera en todas, y modificar simplemente el contenido de la página. Para ello, ya hemos visto en documentos anteriores que podemos utilizar las instrucciones o directivas `include`, `include_once`, `require` o `require_once`.

```
<?php include('cabecera.php'); ?>
    <div class="principal">
    ...
    </div>
    ...
</body>
</html>
```

Cuando incluimos un documento en otro, el funcionamiento es como si copiáramos y pegáramos literalmente todo el código fuente del fichero incluido en el fichero destino. Pero tenemos la ventaja de que, de este modo, si queremos hacer algún cambio (por ejemplo, añadir más opciones al menú de la cabecera), se cambiaría el fichero *cabecera.php* y automáticamente se actualizarían los demás donde se incluye.

Ya hemos explicado en un documento anterior las principales diferencias entre estas cuatro directivas, pero las volvemos a recordar aquí:

- Las directivas **include** e **include_once** provocan una advertencia si no se encuentra el archivo a incluir, mientras que **require** y **require_once** producen un error fatal que cuelga la página (son más restrictivas estas últimas). Usaremos las primeras para incluir contenido que no sea crítico para el funcionamiento de la página, y las segundas para contenido esencial que debe estar presente.
- Las directivas acabadas en *_once* incluyen el documento si no se ha incluido ya previamente con otra directiva include o require. Esto es habitual cuando incluimos ficheros que, a su vez, incluyen otros, y otros... Es posible que en estos casos alguna de las inclusiones afecte a un fichero que pensábamos incluir, y ya está incluido.

Lógicamente, de la misma forma que hemos incluido la cabecera, también podríamos tener otro fichero *pie.php* con el pie de página (cierre de etiquetas, marco legal, copyright, etc.), e incluirlo también en todas las páginas. Por último destacar que, a la hora de incluir páginas dentro de otras, se suele utilizar la siguiente convención:

Si la página incluida no es una página completa por sí misma (sino que necesita de otras para completarse), o es una librería de funciones PHP para poder utilizar en la página donde se incluye, el archivo suele tener extensión *.inc* o *.inc.php*. Así, por ejemplo, la cabecera anterior no sería una página completa por sí misma, y debería haberse llamado *cabecera.inc*. De la misma forma, si hacemos un conjunto de funciones para acceder a una base de datos, y lo queremos incluir es un archivo PHP para usar esas funciones, el archivo debería llamarse, por ejemplo, *libreria.inc*.

> **Ejercicio 2:**
>
> Modifica el ejercicio de libros anterior. Añade en la carpeta un nuevo archivo llamado *cabecera.inc*, con la cabecera de las dos páginas web (el *html*, *head*, algún estilo CSS como color de fondo o tipo de letra, y algún logo en el body que encuentres por Internet relacionado con el mundo de los libros). Después, haz que las otras dos páginas (*form_libros.php* y *result_libros.php*) incluyan la nueva cabecera y quiten las suyas propias, para compartir cabecera.

Ejercicio 3:

Crea en tu carpeta de *ejercicios5* una subcarpeta llamada **formulario**. Dentro, crea un formulario llamado **formulario.php** que permita introducir un nombre, un email y subir un fichero con una imagen a la misma página *formulario.php*.

El fichero de imagen debes guardarlo en una subcarpeta llamada *fotos*, que debes crear previamente. Además, debes comprobar si se ha enviado el formulario o no.

Debes comprobar que se envían todos los campos del formulario; si algún campo está vacío debes redirigir a una página *error.php*, donde mostrarás un mensaje personalizado según el campo que esté vacío. Por ejemplo, si el campo *nombre* está vacío debes mostrar el mensaje "Debes indicar tu nombre". El mensaje personalizado lo puedes enviar mediante un enlace: *error.php?mensaje=Debes indicar tu nombre*. Además, la página de *error.php* mostrará un enlace que permitirá volver a la página formulario.php

Si se han rellenado todos los campos, debes mostrar el valor de los campos enviados e insertar la imagen en la página.