

Prácticas

Estilo: Normal ▼ Tamaño letra: Normal ▼ Interlineado: 1.2 ▼

Desarrollo de Aplicaciones Web

Práctica 7: Administración de un servidor web y PHP 1 (formularios)

1. Objetivos

- Aprender a instalar y configurar la plataforma de desarrollo web XAMPP.
- Aprender los conceptos básicos del lenguaje de programación PHP.
- Aprender a generar código HTML desde PHP.
- Aprender a dividir la estructura de una página web en múltiples ficheros con PHP.
- Aprender a gestionar los datos de un formulario web con PHP.

2. Recursos

¿Cómo se instala y configura XAMPP?

- **XAMPP for Windows** [\(1\)](#): sitio web oficial de esta plataforma software para el desarrollo de aplicaciones web.
- **Apache HTTP Server Documentation** [\(2\)](#): documentación del servidor web Apache disponible en múltiples formatos.

¿Cuál es la sintaxis de PHP? ¿Qué funciones existen?

- **W3Schools** [\(3\)](#): cursos de aprendizaje y guías de referencia de diversas tecnologías empleadas en la programación web. Incluye un tutorial y temas avanzados sobre PHP.
- **PHP Documentation** [\(4\)](#): documentación oficial del lenguaje de programación PHP, disponible para consultar en línea y para descargar.
- **PHP Cheat Sheet** [\(5\)](#): resumen de PHP, sólo incluye algunas de las funciones de PHP: acceso a ficheros, cadenas, expresiones regulares, fechas y formato de fechas.

¿Existe alguna herramienta que me pueda ayudar a escribir el código PHP?

- **Notepad++** [\(6\)](#): editor gratuito de código fuente que soporta varios lenguajes de programación. Entre otras características, posee sintaxis coloreada, envoltura de sintaxis y autocompletado.
- **Sublime Text** [\(7\)](#): excelente editor compatible con múltiples lenguajes de programación y disponible para Windows, OS X y Linux; se puede probar de forma gratuita, pero para un uso continuo hay que adquirir una licencia.
- **Antechinus PHP Editor** [\(8\)](#): potente editor de PHP de pago.

3. ¿Qué tengo que hacer?

En esta práctica tienes que instalar y configurar XAMPP, una plataforma de desarrollo de aplicaciones web que incorpora el servidor web Apache, el sistema gestor de bases de datos MySQL y los lenguajes de programación PHP y Perl [\(9\)](#). A partir de ahora, usaremos XAMPP para desarrollar y probar la aplicación web que estás desarrollando.

Por otro lado, tienes que **dividir la estructura de las páginas de tu sitio web en múltiples ficheros y tienes que utilizar PHP para combinarlos en uno solo**. Detecta la partes que sean comunes a diferentes

páginas de tu sitio web (por ejemplo, la cabecera con el título, la barra de navegación y el pie de página) y ponlas en ficheros independientes. De este modo te puedes crear una plantilla a partir de la cual se genera cualquier página web de tu sitio web.

Además, tienes que programar con PHP las páginas web que reciban los datos enviados a partir de los formularios que has realizado en las prácticas anteriores. Por ahora, estas páginas simplemente tienen que **mostrar los datos recibidos (ya no datos ficticios)** para comprobar que todo funciona correctamente. En próximas prácticas almacenarás estos datos en una base de datos.

En concreto, tienes que modificar o crear las páginas que se indican con un color de relleno oscuro en la Figura 1.

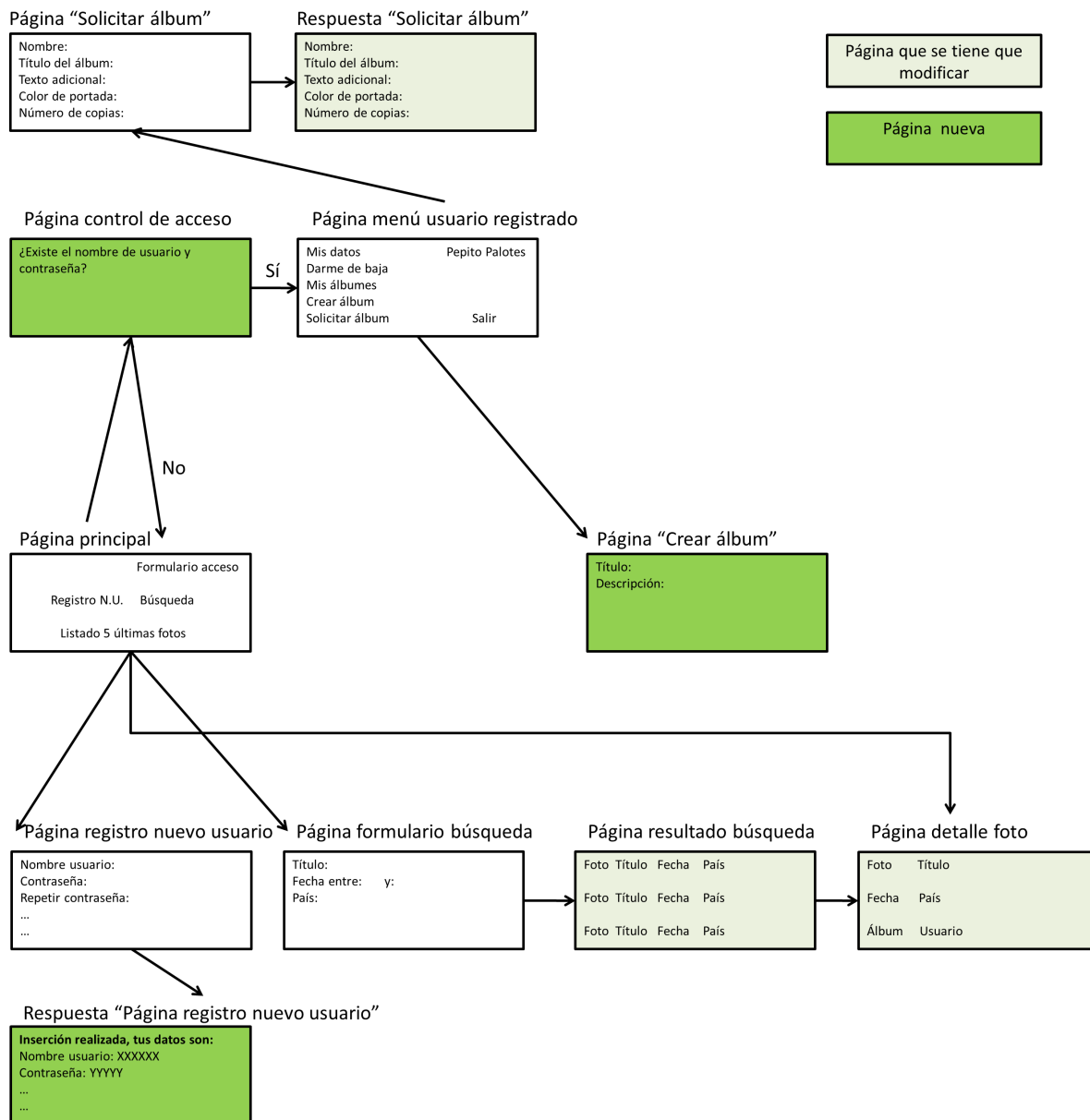


Figura 1: Diagrama de páginas que componen el sitio web

Página control de acceso

No es una página visible. Controla el acceso a la parte privada para los usuarios registrados. Por ahora, se debe limitar el acceso a cuatro posibles usuarios cuyos datos (nombre de usuario, contraseña) están almacenados directamente en esta página (en una próxima práctica se accederá a

una base de datos para consultar los usuarios permitidos). Si el usuario está registrado, mediante una redirección en la parte del servidor se debe mostrar la página con el menú de usuario registrado; si el usuario no está registrado, mediante una redirección en la parte del servidor se debe mostrar la página principal del sitio web con un mensaje de error que explique al usuario lo que ha pasado.

Página “Crear álbum”

Contiene un formulario con los datos necesarios para crear un álbum (título y descripción).

Página “Solicitar álbum”

Además de la tabla de tarifas que ya tenía la página, se muestra una tabla con el coste de un álbum de fotos según los diferentes parámetros (número de páginas, número de fotos, color y resolución), tal como se puede ver en la Figura 2. Esta tabla se tiene que calcular mediante PHP (tienes que sustituir la misma tabla que antes se calculaba mediante JavaScript) y se tienen que aplicar las tarifas que se proporcionaron en una práctica anterior.

Respuesta “Solicitar álbum”

Muestra una confirmación de que se ha registrado la solicitud de un álbum y también muestra el coste del álbum calculado a partir de la solicitud del usuario. Se debe emplear la tabla de tarifas que aparece en la Práctica 3. Como todavía no existen álbumes y fotos en una base de datos, debes emplear unos valores ficticios para el número de páginas y número de fotos del álbum; sin embargo, sí que debes tener en cuenta el resto de parámetros que elija el usuario en el formulario de solicitud.

Respuesta “Página con el formulario de registro como nuevo usuario”

Muestra los datos que el usuario ha introducido en el formulario de registro. Se debe comprobar que se ha escrito algo en el nombre de usuario, en la contraseña y en repetir contraseña, y que contraseña y repetir contraseña coinciden. La validación del resto de campos del formulario se implementará en una próxima práctica. Importante: en esta página no se debe mostrar la fotografía de perfil seleccionada por el usuario, la subida de ficheros al servidor se implementará en una próxima práctica.

Página con el listado resultado de una búsqueda

Además del ejemplo de resultado de una búsqueda que ya contiene (datos estáticos), muestra los datos reales que el usuario ha introducido en el formulario de búsqueda.

Página detalle foto

Muestra el detalle de la foto seleccionada (foto, título, fecha, país, álbum de fotos y usuario al que pertenece). A esta página se puede llegar desde cualquier otra página en la que aparezcan fotos de la aplicación, como por ejemplo, la página principal (las últimas fotos subidas) o la página con el listado resultado de una búsqueda. Por ahora, se deben mostrar los datos de dos fotos distintas que están almacenados directamente en esta página (en una próxima práctica se accederá a una base de datos para obtener los datos reales); la elección de la foto a mostrar dependerá de si el identificador de la foto que se reciba es un número par o impar.

Importante: en esta práctica todavía no debes usar sesiones, eso se verá en una próxima práctica.

Número de páginas	Número de fotos	Blanco y negro		Color	
		150-300 dpi	450-900 dpi	150-300 dpi	450-900 dpi
1	3	0,10	0,16	0,25	0,31
2	6	0,20	0,32	0,50	0,62
3	9	0,30	0,48	0,75	0,93
4	12	0,40	0,64	1,00	1,24
5	15	0,48	0,78	1,23	1,53
6	18	0,56	0,92	1,46	1,82
7	21	0,64	1,06	1,69	2,11
8	24	0,72	1,20	1,92	2,40
9	27	0,80	1,34	2,15	2,69
10	30	0,88	1,48	2,38	2,98
11	33	0,96	1,62	2,61	3,27
12	36	1,03	1,75	2,83	3,55
13	39	1,10	1,88	3,05	3,83
14	42	1,17	2,01	3,27	4,11
15	45	1,24	2,14	3,49	4,39

Figura 2: Tabla con posibles costes de un álbum

Importante: el código que programes debe ser eficaz, eficiente y modular.

3.1. Recordatorio

La parte privada de la aplicación y su integración con la parte pública la puedes plantear de varias formas. Dos formas típicas son:

Separada

La parte privada es completamente independiente de la parte pública, posee su propio menú o barra de navegación e incluso puede poseer su propio estilo visual (CSS). Evidentemente, debe existir una opción en el menú o barra de navegación que permita pasar de la parte pública a la parte privada y viceversa.

Integrada

La parte privada se integra como una opción más en el menú o barra de navegación de la parte pública. La parte privada aparece como un apartado más de la parte pública, que sólo está disponible cuando el usuario se ha identificado.

En la realización de esta práctica puedes aplicar cualquiera de estas dos estrategias o cualquiera similar.

3.2. Desarrollo de un miniframework

¿Quieres intentar obtener la máxima nota en esta y en las próximas prácticas? En esta asignatura no puedes usar un *framework*, pero sí que puedes construir y usar tu propio *framework*. Para ello:

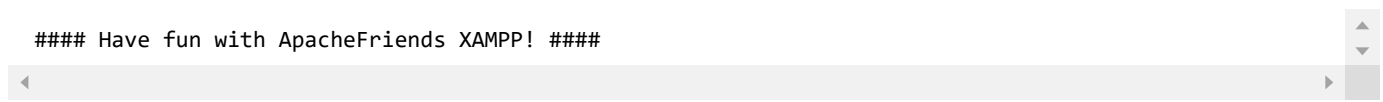
1. Desarrolla un enrutador para controlar la ejecución de tu práctica. Todo el tráfico debe pasar por el fichero `index.php`, que según la petición recibida, dirija el tráfico a la parte de la aplicación adecuada.
2. Controla que solo se pueda mostrar un fichero del sitio web cuando sea llamado desde `index.php`.
3. Oculta el fichero `index.php` de las direcciones. Si todo el tráfico se dirige al mismo fichero, no necesitas que aparezca el fichero en el URL.
4. Utiliza URL amigables (en inglés, *clean URLs* [\(10\)](#) o *user-friendly URLs*).
5. Aplica el patrón de arquitectura de software modelo-vista-controlador [\(11\)](#) (MVC). Este patrón permite separar los datos de la lógica de negocio y de la interfaz de la aplicación, lo que facilita el desarrollo y reutilización del código.

4. ¿Cómo lo hago?

4.1. XAMPP

XAMPP es una aplicación portable que no necesita instalación, aunque puede ser instalada con un instalador si así se desea. Al ser una aplicación portable se pueden trasladar de un ordenador a otro sin perder datos y sin tener que realizar complicadas reconfiguraciones. En la página de XAMPP hay disponibles varias versiones para descargar: **Installer**, **ZIP archive** y **Selfextracting 7-ZIP archive**. Para el propósito de lograr una instalación portable podemos emplear cualquiera de las dos últimas.

Una vez descomprimido (se recomienda instalarlo en la raíz de una unidad, por ejemplo `C:\xampp` o `D:\xampp`), se tiene que ejecutar desde la línea de comandos el fichero `setup_xampp.bat` para configurar correctamente los directorios de ejecución de cada aplicación. Si la configuración se realiza correctamente, debe aparecer en pantalla:



```
#### Have fun with ApacheFriends XAMPP! ####
```

Una vez configurado, se deben iniciar el servidor web Apache y el sistema gestor de bases de datos MySQL. Para ello existen varias formas, pero las dos más comunes son:

- Ejecutar desde la línea de comandos el fichero `xampp_start.exe`: esta ventana no se debe cerrar mientras se esté utilizando XAMPP, tal como indica el mensaje que se muestra al ejecutar este fichero (ver Figura [3](#)). Se debe emplear `xampp_stop.exe` para detener los servicios de Apache y MySQL, si los servicios se detienen correctamente aparecerá un mensaje similar al mostrado en la Figura [4](#).
- Ejecutar el programa `xampp-control.exe`: inicia un panel de control de XAMPP (ver Figura [5](#)) que se quedará activo en la barra de tareas hasta que se cierre pulsando el botón **Quit**. Para iniciar cada servicio simplemente se tiene que pulsar sobre el botón **Start** correspondiente. Para detener cada servicio pulsar sobre el botón **Stop** correspondiente.

```

C:\WINDOWS\system32\cmd.exe

C:\xampp>xampp_start.exe
Diese Eingabeforderung nicht waehrend des Runnings beenden ...
Zum stoppen bitte die xampp_stop benutzen!
Please do not close this window while running ...
Use the xampp_stop for shutdown!

Please wait [Bitte warten] ...

### APACHE + MYSQL IS STARTING NOW ###

C:\xampp>

```

Figura 3: Inicio de XAMPP desde la línea de comandos

```

C:\WINDOWS\system32\cmd.exe

C:\xampp>xampp_stop
MySQL stop ...
Apache killed ...
apache.exe (6116)
apache.exe (3708)
Apache pid delete ... SHUTDOWN COMPLETE!

C:\xampp>

```

Figura 4: Detención de XAMPP desde la línea de comandos

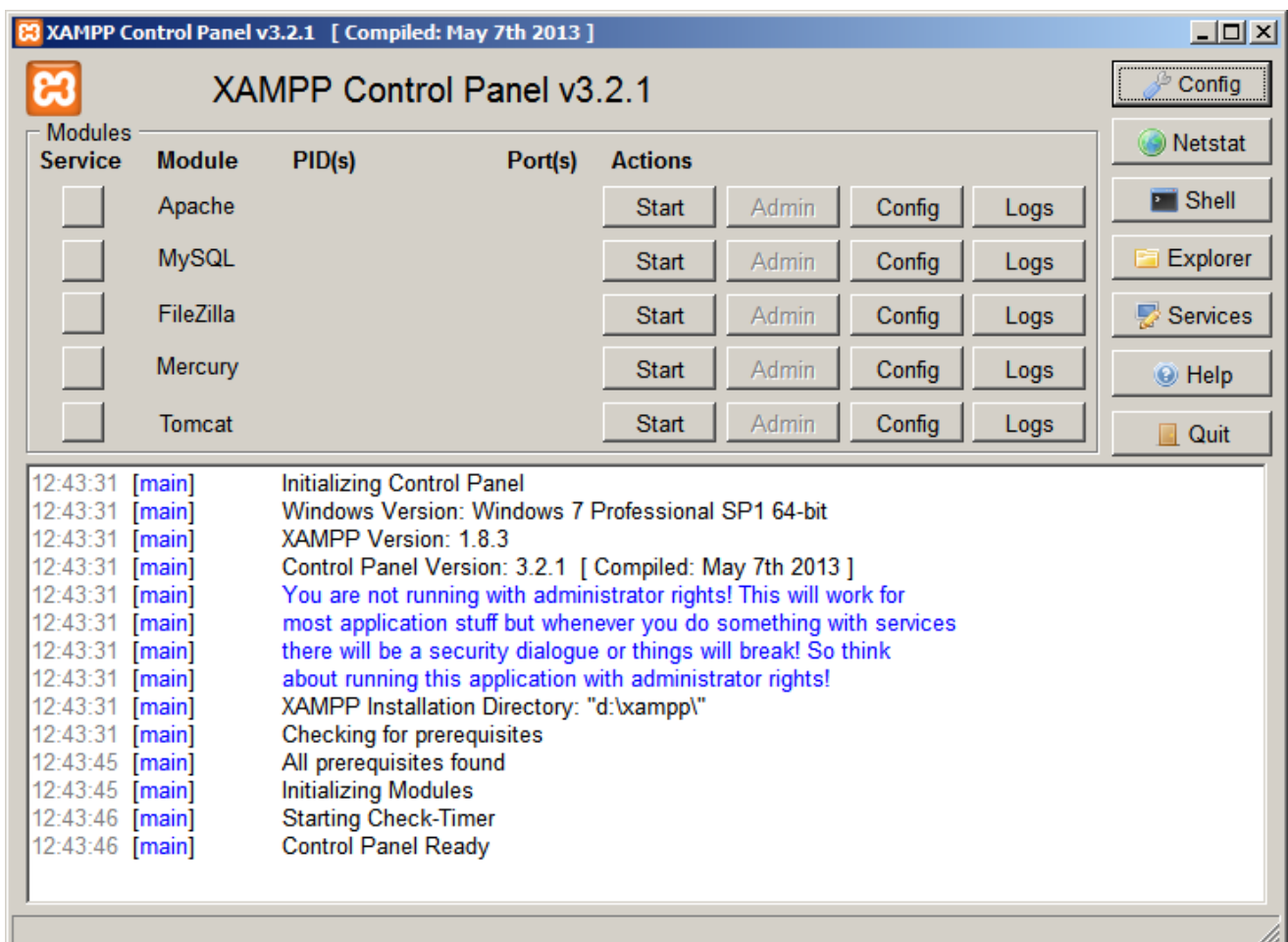


Figura 5: Panel de control de XAMPP

Estas dos formas de iniciar los servidores son equivalentes, por lo que, o se emplea una forma o la otra. Si se emplea la primera forma con `xampp_start.exe`, al iniciar el panel de control de XAMPP deberían de aparecer los puertos que se están utilizando y los identificadores de proceso PID de los servicios Apache y MySQL.

4.2. Alojamiento de un sitio web

El directorio `\xampp\htdocs` es el directorio de publicación web: todo lo que se almacene en este directorio es accesible a través de la dirección `http://localhost/` a través de un navegador.

En algunas ocasiones, tener que publicar un sitio web en `\xampp\htdocs` puede plantear problemas o puede ser imposible. El módulo `mod_alias` de Apache [\(12\)](#) permite la publicación de un sitio web en cualquier parte del sistema de ficheros. Para ello, es necesario modificar el fichero de configuración de Apache `httpd.conf` que se encuentra en el directorio `\xampp\apache\conf`. La directiva `Alias` permite definir el mapeo [\(13\)](#) entre una URL y un directorio del sistema de ficheros. En el siguiente ejemplo, todo el tráfico que llegue a `http://localhost/practica` será dirigido al directorio del sistema de ficheros `C:/Users/Sergio/Mis documentos/practica`:

```
Alias /practica "C:/Users/Sergio/Mis documentos/practica"

<Directory "C:/Users/Sergio/Mis documentos/practica">
    Options Indexes FollowSymLinks Includes ExecCGI
    AllowOverride All
    Require all granted
</Directory>
```

La directiva `Directory` [\(14\)](#) permite agrupar un conjunto de directivas que se aplican a un directorio del sistema de ficheros.

4.3. Creación de una plantilla

Un sitio web correcto debe mantener una coherencia, tanto visual como de estructura y contenido, entre todas sus páginas web. Para lograrlo, lo normal es trabajar con una plantilla a partir de la cual se crean todas las páginas web. Cuando se quiera hacer un cambio, se modifica la plantilla y todas las páginas web se verán afectadas por el cambio.

Otra técnica, que suelen emplear la mayoría de los gestores de contenidos, es trabajar con una única página web que gestiona la visualización de todo el contenido del sitio web. Para ello, todos los enlaces del sitio web tienen como destino a esa única página y mediante un parámetro se indica el contenido que se quiere visualizar.

En tu práctica puedes emplear la primera técnica de la plantilla, aunque si quieres puedes emplear el segundo método de una única página (un poco más complicado).

En PHP, para incluir en un fichero el contenido de otro fichero se emplean dos funciones muy parecidas, `require(fichero)` e `include(fichero)`:

- `require(fichero)`: si el fichero no existe, se produce un mensaje de error y finaliza la ejecución.
- `include(fichero)`: si el fichero no existe, se produce un mensaje de advertencia y continúa la ejecución
- Además, existen `require_once(fichero)` e `include_once(fichero)`: sólo incluyen el fichero indicado una vez, por lo que evitan problemas de redefinición de funciones, reasignación de variables, etc.

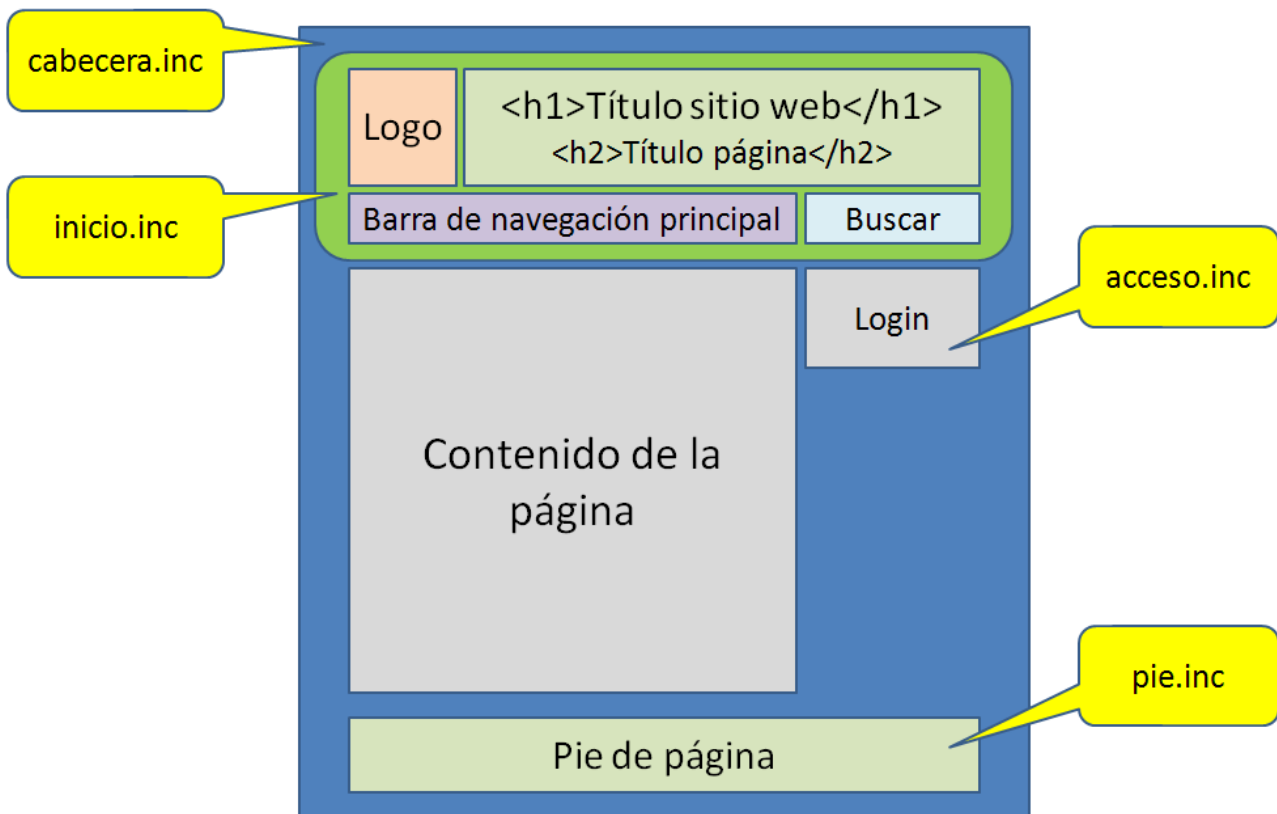


Figura 6: Plantilla de una página web

Por ejemplo, en la Figura 6 se muestra una plantilla sencilla para un sitio web, donde la estructura y contenido de todas las páginas se ha dividido en cinco partes: la cabecera (que contiene el DOCTYPE y el `<head>` de la página), el inicio (que contiene el logotipo, el título del sitio web, la barra de navegación principal y el cuadro de buscar), el control de acceso a la parte privada (que quizás sólo aparezca en la página principal), el contenido principal de la página y el pie de la página. A continuación se muestra el código de una posible página con esta estructura:

```
<?php
// Título de la página, se muestra en <title> y en el cuerpo de la página con <h2>
$title = "El título de esta página";
// Declaración de DOCTYPE, <html>, <head>, <meta>, <link>, etc.
// Contiene <title><?php echo $title; ?></title>
require_once("cabecera.inc");

// Inicio de la página
// Contiene <body>
// Muestra logotipo, título del sitio web, barra de navegación principal,
// cuadro de buscar, etc.
// Contiene <h2><?php echo $title; ?></h2>, con <h1> está marcado el título
// del sitio web
require_once("inicio.inc");

// Acceso a la parte privada de la aplicación (login)
// Sólo aparece en la página principal
require_once("acceso.inc");

// El contenido principal de la página
?>

<main>
<p>
Esto es lo que cambiará de una página a otra.
</p>
```



```

</main>

<?php
// El pie de la página: copyright, declaración legal, dirección de correo, etc.
// Contiene </body></html>
require_once("pie.inc");
?>

```

En el ejemplo anterior se emplea `require_once()` para incluir los ficheros `cabecera.inc`, `inicio.inc`, `acceso.inc` y `pie.inc`.

Cada uno de los ficheros incluidos puede contener código HTML y código PHP, y también puede incluir otros ficheros. A modo de ejemplo se muestra el contenido del fichero `cabecera.inc`:

```

<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title><?php echo $title; ?></title>
</head>

```

4.4. Gestión de formularios

La gestión de formularios en PHP se realiza mediante las variables globales predefinidas `$_GET` (cuando los datos se envían mediante HTTP GET) y `$_POST` (cuando los datos se envían mediante HTTP POST). Ambas son de tipo `superglobal`, por lo que se pueden emplear en cualquier contexto sin tener que declararlas previamente con `global`. Además, ambas son arrays asociativos por lo que se accede a su contenido a través del nombre del control del formulario que queremos acceder. También se puede emplear la variable global predefinida `$_REQUEST`, que contiene la unión de las variables `$_GET`, `$_POST` y `$_COOKIE` (permite gestionar las *cookies*, tanto para leer como escribir), pero se desaconseja su uso porque puede originar errores.

Por ejemplo, la siguiente página web contiene un formulario con tres controles:

```

<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Formulario</title>
</head>
<body>
<form action="formulario.php?id=123" method="post">
<p>
Nombre: <input type="text" name="nombre" />
<br />
Apellidos: <input type="text" name="apellidos" />
<br />
Deportes:
<select name="deportes[]" multiple="multiple">
<option value="1">Baloncesto</option>
<option value="2">Fútbol</option>
<option value="3">Paddle</option>
<option value="4">Tenis</option>
</select>
<br />
<input type="submit" value="Enviar" />
<input type="reset" value="Borrar" />
</p>
</form>

```

```
</body>
</html>
```

Los controles están etiquetados como nombre, apellidos y deportes[]. Este último control tiene los corchetes para que desde PHP se pueda procesar correctamente como un array de valores, ya que la lista desplegable tiene el atributo `multiple` y permite seleccionar varios valores a la vez. El formulario se envía mediante HTTP POST a la página `formulario.php`.

La página `formulario.php` simplemente muestra los datos recibidos, para ello emplea la función `print_r()` que permite visualizar todo el contenido de un array de una forma legible; también se emplea `$_POST` para mostrar directamente el contenido de los controles nombre y apellidos:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8" />
<title>Respuesta formulario</title>
</head>
<body>
<pre>
<?php
echo "Contenido de \$_GET:\n";
print_r($_GET);

echo "\n";
echo "Contenido de \$_POST:\n";
print_r($_POST);

echo "\n";
echo "Contenido de \$_REQUEST:\n";
print_r($_REQUEST);
?>
</pre>
<p>
Nombre: <b><?php echo $_POST["nombre"];?></b>
<br />
Apellidos: <b><?php echo $_POST["apellidos"];?></b>
</p>
</body>
</html>
```

Recuerda que para que estas páginas funcionen se tienen que almacenar dentro del directorio `\xampp\htdocs`; todo lo que se almacene en este directorio es accesible a través de la dirección `http://localhost/` a través de un navegador. Si se abre de forma local una página web en el navegador (aparece algo como `C:\xampp\htdocs\formulario.html` en la barra de direcciones), la página no será ejecutada por el servidor web y el navegador recibirá el código PHP. Recuerda también que las páginas que contengan código PHP tienen que tener la extensión `.php` [\(15\)](#).

4.5. Variables superglobales

`$_GET` y `$_POST` son variables superglobales [\(16\)](#) de PHP. Las variables superglobales son variables predefinidas que están disponibles en todos los ámbitos a lo largo del código. Por tanto, no es necesario emplear `global $variable;` para acceder a ellas dentro de las funciones o métodos.

Las variables superglobales son arrays asociativos con el siguiente contenido:

- `$GLOBALS`: variables disponibles en el ámbito global.
- `$_SERVER`: información del entorno del servidor y de ejecución.
- `$_GET`: variables HTTP GET.

- `$_POST`: variables HTTP POST.
- `$_FILES`: variables de subida de ficheros HTTP.
- `$_COOKIE`: variables almacenadas en las cookies HTTP.
- `$_SESSION`: variables de sesión.
- `$_REQUEST`: contiene el contenido de `$_GET`, `$_POST` y `$_COOKIE`.
- `$_ENV`: variables de entorno.

4.6. Redirección

Para realizar una redirección en el lado del servidor se emplea la función `header()` de PHP que permite enviar encabezados HTTP directamente al navegador. La redirección se realiza enviando el encabezado `Location`: con la URL de la página a la que se quiere redirigir.

La llamada a la función `header()` se tiene que hacer antes de que la página haya generado cualquier resultado. Un simple espacio en blanco o un salto de línea ocasionará un error: el BOM del formato de codificación UTF-8 producirá un error, por lo que se recomienda almacenar las páginas PHP con el formato “UTF-8 sin BOM”.

En el siguiente fragmento de código se muestra cómo usar la función `header()` para realizar una redirección:

```
<?php
/* Redirecciona a una página diferente que se encuentra en el directorio actual */
$host = $_SERVER['HTTP_HOST'];
$uri = rtrim(dirname($_SERVER['PHP_SELF']), '/\\');
$extra = 'paginaDestino.php';
header("Location: http://$host$uri/$extra");
exit;
?>
```

HTTP 1.1 [\(17\)](#) obliga a utilizar URL absolutas cuando se realiza una redirección. Para obtener una URL absoluta se emplean `$_SERVER['HTTP_HOST']` que devuelve el nombre del servidor, `$_SERVER['PHP_SELF']` que devuelve la ruta relativa al fichero actual que se está ejecutando (por ejemplo, si el fichero se encuentra en `http://ejemplo.com/directorio/test.php` devolverá `/directorio/test.php`) y `dirname()` que devuelve la parte correspondiente al directorio indicado en una ruta a un fichero.

En la redirección se pueden pasar parámetros, como en cualquier URL. Si quieres pasar varios parámetros, la cadena de consulta la puedes construir a mano o puedes usar la función `http_build_query()` [\(18\)](#).

4.7. Construcción de un miniframework

¿Qué es un framework? ¿Cómo funciona un framework? Vamos a ver lo que se dice en la documentación de CodeIgniter [\(19\)](#), un framework de PHP muy popular:

Welcome to CodeIgniter [\(20\)](#)

CodeIgniter is an Application Development Framework - a toolkit - for people who build web sites using PHP. Its goal is to enable you to develop projects much faster than you could if you were writing code from scratch, by providing a rich set of libraries for commonly needed tasks, as well as a simple interface and logical structure to access these libraries. CodeIgniter lets you creatively focus on your project by minimizing the amount of code needed for a given task.

Un framework permite desarrollar aplicaciones de una forma más rápida y con menos errores, ya que facilita la escritura y el mantenimiento del código.

En la Figura 7 se muestra el flujo de ejecución de una petición en CodeIgniter [\(21\)](#):

1. The index.php serves as the front controller, initializing the base resources needed to run CodeIgniter.
2. The Router examines the HTTP request to determine what should be done with it.
3. If a cache file exists, it is sent directly to the browser, bypassing the normal system execution.
4. Security. Before the application controller is loaded, the HTTP request and any user submitted data is filtered for security.
5. The Controller loads the model, core libraries, helpers, and any other resources needed to process the specific request.
6. The finalized View is rendered then sent to the web browser to be seen. If caching is enabled, the view is cached first so that on subsequent requests it can be served.

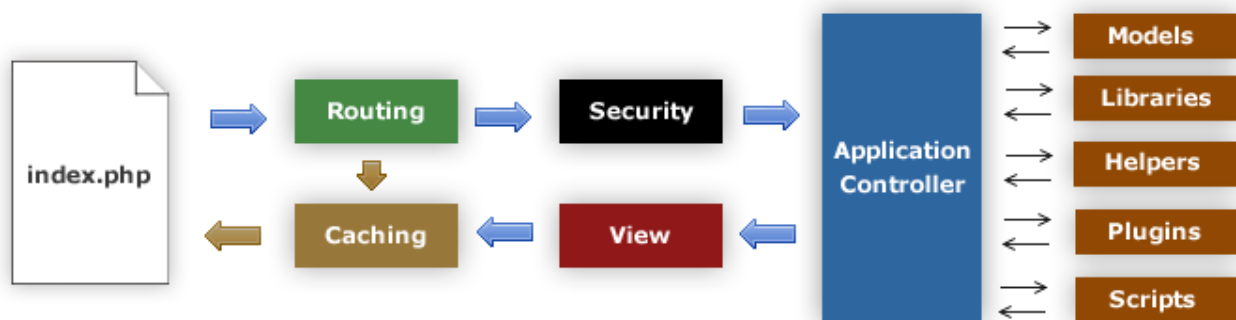


Figura 7: Flujo de ejecución de CodeIgniter

En tu miniframework no debes implementar las mismas características que tiene CodeIgniter, pero el flujo de ejecución sí que debe ser similar. Por ejemplo, la cache, la capa de seguridad, los helpers y los plugins son temas avanzados que es mejor no considerar en una primera versión de un framework.

Lo que sí que es básico es que utilices el patrón MVC, que en la documentación de CodeIgniter [\(22\)](#) se define de la siguiente forma:

- The **Model** represents your data structures. Typically your model classes will contain functions that help you retrieve, insert, and update information in your database.
- The **View** is the information that is being presented to a user. A View will normally be a web page, but in CodeIgniter, a view can also be a page fragment like a header or footer. It can also be an RSS page, or any other type of “page”.
- The **Controller** serves as an intermediary between the Model, the View, and any other resources needed to process the HTTP request and generate a web page.

CodeIgniter, y la mayoría de los frameworks, impone un patrón para las URL [\(23\)](#):

The segments in the URL, in following with the Model-View-Controller approach, usually represent:

example.com/class/function/ID

1. The first segment represents the controller **class** that should be invoked.
2. The second segment represents the class **function**, or method, that should be called.
3. The third, and any additional segments, represent the **ID and any variables** that will be passed to the controller.

Este esquema de URL se puede aplicar con o sin programación orientada a objetos.

En la documentación de CodeIgniter [\(24\)](#) también se explica cómo eliminar el fichero `index.php` de las URL:

By default, the `index.php` file will be included in your URLs:

`example.com/index.php/news/article/my_article`

If your Apache server has `mod_rewrite` enabled, you can easily remove this file by using a `.htaccess` file with some simple rules. Here is an example of such a file, using the “negative” method in which everything is redirected except the specified items:

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php/$1 [L]
```

In the above example, any HTTP request other than those for existing directories and existing files is treated as a request for your `index.php` file.

5. Recomendaciones

Siempre se debe proporcionar una indicación (*feedback*) de lo que está pasando al usuario. Si se produce un error, se debe mostrar un mensaje de error significativo. Por ejemplo, cuando un usuario intenté acceder (*login*) y sus datos no sean correctos, se debe mostrar un mensaje de error.

Muy importante: en la realización de las prácticas **no debes** emplear un *framework* de PHP. En el momento de escribir esto, aparecen 24 *frameworks* en “Comparison of web frameworks - PHP” [\(25\)](#), tal como se puede ver en la Figura [8](#), y seguro que existen muchos más que ni aparecen. Antes de elegir y aprender un *framework*, hay que conocer bien la tecnología sobre la que se construye. Ese es el objetivo de esta asignatura.

PHP [[edit](#)]See also: [PHP](#)

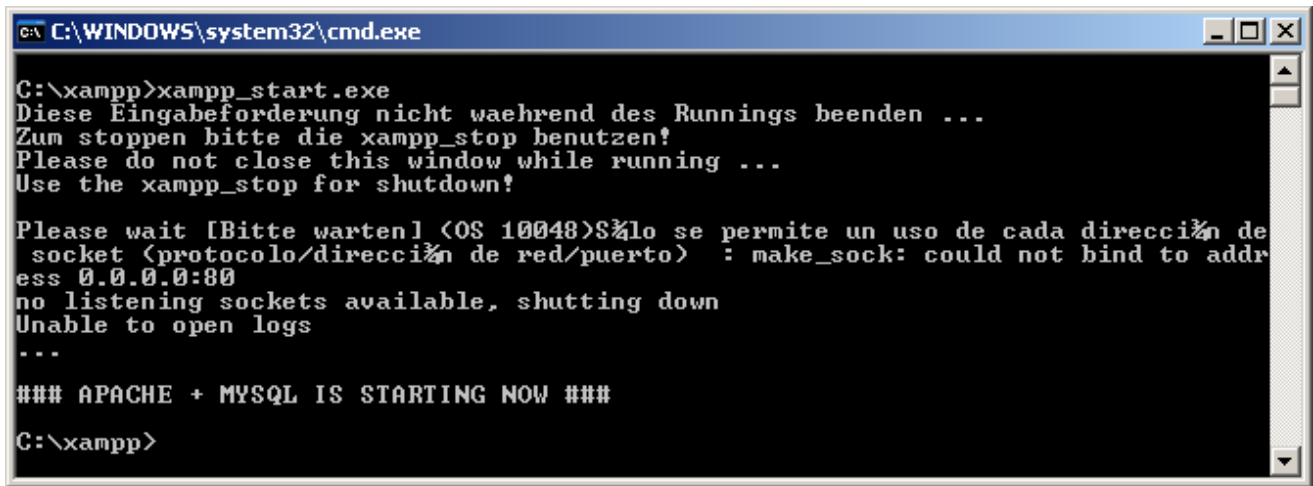
Project	Start date	Current stable version	Release date	License
Agavi	2005-05	1.0.8 ^[17]	2015-06-29	LGPL
CakePHP	2005-08	3.6.11 ^[18]	2018-09-02[±]	MIT
CodeIgniter	2006-02-28	3.1.9 ^[19]	2018-06-12	MIT
Drupal	2000-05-18	8.4.4 ^[20]	2018-01-03	GPLv2
Fat-Free	2009-09	3.6.4 ^[21]	2018-04-19	GPLv3
FuelPHP	2011-08	1.8 ^[22]	2016-04-09	MIT
Gyroscope	2008-11-20	8.8.0	2016-04-17	BSD
Jamroom	2003-07-28	6.1.0 ^[23]	2017-08-30	MPL
Kajona	2006	6.2 ^[24]	2017-06-08	LGPLv2
Kohana	2007-07	3.3.5 ^[25]	2016-03-10	BSD
Laravel	2011-07-31	5.6.29 ^[26]	2018-07-31	MIT
Li3 (Lithium)	2009-10	1.1.0 ^[27]	2017-04-23	BSD
Nette Framework	2006-01 ^[28]	2.4.0 ^[29]	2016-05-03	New BSD, GPLv2, GPLv3 ^[30]
Phalcon	2012-11-14	3.2.3 ^[31]	2017-10-12	BSD
Pop PHP	2012-03-19	3.6.1 ^[32]	2017-09-14	New BSD
PRADO	2004-01	3.3.2 ^[33]	2016-08-23	New BSD ^[34]
Silex	2011-09	2.0.0 ^[35]	2016-05-18	MIT
SilverStripe	2007-02-03	3.6.1 ^[36]	2017-06-27	BSD
Smart.Framework	2015-02-01	2.3.7.2 ^[37]	2016-09-27	BSD
Symfony	2005-10	4.1.4 ^[38]	2018-08-28[±]	MIT
TwistPHP	2014-07	3.0.5 ^[39]	2017-01-11	GPLv3
TYPO3 Flow	2011-10	3.3.4 ^[40]	2016-09-29	LGPLv3
Yii	2008-12-03	2.0.14 ^[41]	2018-02-19	New BSD
Zend Framework	2006-03	3.0.0 ^[42]	2016-06-28	New BSD

Figura 8: Lista de frameworks de PHP disponible en la Wikipedia

Ten cuidado con el BOM (*Byte Order Mark*).⁽²⁶⁾ de UTF-8. Como se explica en “Problemas de visualización provocados por BOM en UTF-8”⁽²⁷⁾, puede aparecer una línea adicional o caracteres no deseados en la parte superior de una página web o un archivo incluido. En PHP el problema ocurre cuando se construye una página web incluyendo varios ficheros. La solución es sencilla: utiliza UTF-8 sin BOM.

En el artículo “Install portable WAMP (Windows, Apache, MySQL, PHP)”⁽²⁸⁾, se explica cómo realizar una instalación portable de XAMPP, de forma que la puedas mover de un ordenador a otro sin problemas (por ejemplo, hacer una instalación en una memoria USB). Si tienes algún problema con XAMPP, consulta “XAMPP for Windows FAQ”⁽²⁹⁾.

Cuando instales XAMPP en un ordenador, el principal problema que puedes tener es que ya exista algún programa que esté enlazado a los mismos puertos que utiliza XAMPP, como puede ser un servidor web o una instalación previa de MySQL. Si existe este problema, cuando intentes iniciar XAMPP aparecerá un error como el mostrado en la Figura 9 donde se indica que no es posible enlazarse al puerto 80 porque ya está ocupado.



```
C:\WINDOWS\system32\cmd.exe

C:\xampp>xampp_start.exe
Diese Eingabeforderung nicht waehrend des Runnings beenden ...
Zum stoppen bitte die xampp_stop benutzen!
Please do not close this window while running ...
Use the xampp_stop for shutdown!

Please wait [Bitte warten] (OS 10048)Sólo se permite un uso de cada dirección de
socket (protocolo/dirección de red/puerto) : make_sock: could not bind to addr
ess 0.0.0.0:80
no listening sockets available, shutting down
Unable to open logs
...

### APACHE + MYSQL IS STARTING NOW ###

C:\xampp>
```

Figura 9: Error al inicio de XAMPP desde la línea de comandos

Para comprobar si los puertos están ocupados puedes emplear varios métodos. El propio panel de control de XAMPP (ver Figura 5) dispone de una opción llamada **Netstat** que muestra el estado de todos los puertos utilizados en el sistema. Por ejemplo, en la Figura 10 se puede ver que los puertos 80 (HTTP) y 443 (HTTPS) que necesita el servidor web Apache y el puerto 3306 que necesita MySQL no están utilizados, están disponibles. Cuando se activan estos dos servicios, si se vuelve a verificar el estado de los puertos se comprueba que en los puertos 80 y 443 aparece el proceso httpd.exe (Apache) y en el puerto 3306 el proceso mysqld.exe.

Address	Port	PID	Name
0.0.0.0	135	868	svchost.exe
192.168.1.3	139	4	System
0.0.0.0	445	4	System
0.0.0.0	554	5100	wmpnetwk.exe
0.0.0.0	2869	4	System
0.0.0.0	5357	4	System
0.0.0.0	10243	4	System
127.0.0.1	12025	1556	AvastSvc.exe
127.0.0.1	12080	1556	AvastSvc.exe
127.0.0.1	12110	1556	AvastSvc.exe
127.0.0.1	12119	1556	AvastSvc.exe
127.0.0.1	12143	1556	AvastSvc.exe
127.0.0.1	12465	1556	AvastSvc.exe
127.0.0.1	12563	1556	AvastSvc.exe
127.0.0.1	12993	1556	AvastSvc.exe
127.0.0.1	12995	1556	AvastSvc.exe
0.0.0.0	17500	3840	Dropbox.exe
127.0.0.1	19872	3840	Dropbox.exe
127.0.0.1	27275	1556	AvastSvc.exe
0.0.0.0	49152	532	wininit.exe
0.0.0.0	49153	1004	svchost.exe
0.0.0.0	49154	356	svchost.exe
0.0.0.0	49155	628	lsass.exe
0.0.0.0	49159	592	services.exe
127.0.0.1	49210	3840	Dropbox.exe
127.0.0.1	56408	3840	Dropbox.exe
192.168.1.3	56409	1556	AvastSvc.exe
192.168.1.3	57039	1556	AvastSvc.exe
192.168.1.3	57136	4432	chrome.exe
192.168.1.3	57137	4432	chrome.exe
192.168.1.3	57144	3840	Dropbox.exe
192.168.1.3	57145	3840	Dropbox.exe
192.168.1.3	57148	3840	Dropbox.exe

Figura 10: Estado de los puertos según la opción Netstat de XAMPP Control Panel

Address	Port	PID	Name
0.0.0.0	80	12000	httpd.exe
0.0.0.0	135	868	svchost.exe
192.168.1.3	139	4	System
0.0.0.0	443	12000	httpd.exe
0.0.0.0	445	4	System
0.0.0.0	554	5100	wmpnetwk.exe
0.0.0.0	2869	4	System
0.0.0.0	3306	10364	mysql.exe
0.0.0.0	5357	4	System
0.0.0.0	10243	4	System
127.0.0.1	12025	1556	AvastSvc.exe
127.0.0.1	12080	1556	AvastSvc.exe
127.0.0.1	12110	1556	AvastSvc.exe
127.0.0.1	12119	1556	AvastSvc.exe
127.0.0.1	12143	1556	AvastSvc.exe
127.0.0.1	12465	1556	AvastSvc.exe
127.0.0.1	12563	1556	AvastSvc.exe
127.0.0.1	12993	1556	AvastSvc.exe
127.0.0.1	12995	1556	AvastSvc.exe
0.0.0.0	17500	3840	Dropbox.exe
127.0.0.1	19872	3840	Dropbox.exe
127.0.0.1	27275	1556	AvastSvc.exe
0.0.0.0	49152	532	wininit.exe
0.0.0.0	49153	1004	svchost.exe
0.0.0.0	49154	356	svchost.exe
0.0.0.0	49155	628	lsass.exe
0.0.0.0	49159	592	services.exe
127.0.0.1	49210	3840	Dropbox.exe
127.0.0.1	56408	3840	Dropbox.exe
192.168.1.3	56409	1556	AvastSvc.exe
192.168.1.3	57157	1556	AvastSvc.exe
192.168.1.3	57165	5528	MpCmdRun.exe
0.0.0.0	62961	1724	spoolsv.exe

Figura 11: Estado de los puertos según la opción Netstat de XAMPP Control Panel

Si se cambian los puertos que emplea Apache, se pueden tener dos o más servidores web en ejecución en el mismo ordenador al mismo tiempo. Para ello, es necesario modificar el fichero `httpd.conf` que se encuentra en el directorio `\xampp\apache\conf` y el fichero `httpd-ssl.conf` que se encuentra en `\xampp\apache\conf\extra`.

En el fichero `httpd.conf` buscamos las siguientes líneas y cambiamos el puerto 80 por otro puerto (normalmente se suele poner 8080):

```
Listen 80
ServerName localhost:80
```

En el fichero `httpd-ssl.conf` buscamos las siguientes líneas y cambiamos el puerto 443 por otro puerto (por ejemplo, 4443):

```
Listen 443
ServerName localhost:443
```

Para comprobar el estado de los puertos, también podemos utilizar el comando netstat del sistema operativo Microsoft Windows (con los parámetros -a -b -n), que muestra el estado de todas las conexiones de red. Por ejemplo, en la Figura 12 podemos observar que por un lado está enlazado el proceso inetinfo.exe (el servidor web Internet Information Server de Microsoft) a los puertos 80 y 443, y por otro lado apache.exe está enlazado a los puertos 4443 y 8080 (se han cambiado los puertos de Apache porque ya estaban en uso por Internet Information Server).

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\slujan>netstat -a -b -n

Conexiones activas

Proto  Dirección local          Dirección remota         Estado      PID
-----
TCP    0.0.0.0:80                0.0.0.0:0                LISTENING   6036
[inetinfo.exe]

TCP    0.0.0.0:135               0.0.0.0:0                LISTENING   880
c:\windows\system32\WS2_32.dll
C:\WINDOWS\system32\RPCRT4.dll
c:\windows\system32\RPCSS.dll
C:\WINDOWS\system32\svchost.exe
C:\WINDOWS\system32\ADVAPI32.dll
[svchost.exe]

TCP    0.0.0.0:443               0.0.0.0:0                LISTENING   6036
[inetinfo.exe]

TCP    0.0.0.0:445               0.0.0.0:0                LISTENING    4
[System]

TCP    0.0.0.0:2456              0.0.0.0:0                LISTENING   6036
[inetinfo.exe]

TCP    0.0.0.0:3306              0.0.0.0:0                LISTENING   2056
[mysqld.exe]

TCP    0.0.0.0:4443              0.0.0.0:0                LISTENING   3336
[apache.exe]

TCP    0.0.0.0:8080              0.0.0.0:0                LISTENING   3336
[apache.exe]
```

Figura 12: Estado de los puertos según netstat

¡Cuidado! Skype puede utilizar los puertos 80 y 443 para las conexiones entrantes [\(30\)](#), así que puedes tener problemas con XAMPP y Skype.

Recuerda que XAMPP es una plataforma de desarrollo y no está orientada a ser usada en producción, ya que no está configurada para obtener un máximo rendimiento y puede tener problemas de seguridad.

El manual de PHP te lo puedes descargar en diferentes formatos de su sitio web [\(31\)](#) para tenerlo siempre a mano y poder hacer las búsquedas de información rápidamente. También puedes acceder a través de Internet a la ayuda de cualquier función de PHP escribiendo el nombre de la función a continuación de la URL <http://php.net/>. Por ejemplo, <http://php.net/header> muestra la ayuda de la función `header()`.

Si estás escribiendo correctamente el código XHTML, habrás puesto al principio de cada página la declaración de XML [\(32\)](#):

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

Esta declaración puede ocasionar un error, ya que en PHP también se emplean los símbolos `<?` para indicar el inicio del código PHP [\(33\)](#), por lo que el intérprete de PHP toma la declaración de XML como un fragmento de código PHP y muestra el siguiente mensaje de error al intentar interpretarlo:

```
Parse error: syntax error, unexpected T_STRING in C:\xampp\htdocs\index.php on line 1
```

Este problema se puede solucionar de dos formas. La primera supone modificar el fichero `php.ini` de configuración del intérprete de PHP para desactivar el uso de `<?`. En el caso de XAMPP, este fichero está alojado en el directorio `\xampp\apache\bin` y se tiene que poner a `Off` el parámetro de configuración `short_open_tag`:

```
; Allow the <? tag. Otherwise, only <?php and <script> tags are recognized.
; NOTE: Using short tags should be avoided when developing applications or
; libraries that are meant for redistribution, or deployment on PHP
; servers which are not under your control, because short tags may not
; be supported on the target server. For portable, redistributable code,
; be sure not to use short tags.
short_open_tag = Off
```

Recuerda: cuando el parámetro `short_open_tag` está desactivado, la forma corta de realizar un `echo()` mediante `<?=expresion?>` no funciona en todas las versiones de PHP [\(34\)](#):

- En versiones anteriores a PHP 5.4.0, `short_open_tag` con valor `Off` impedía el uso `<?=expresion?>`.
- A partir de PHP 5.4.0, `<?=expresion?>` siempre está disponible.

Si se desactiva el uso de `<?`, el inicio del código PHP se tiene que indicar con `<?php`.

La otra solución consiste en escribir la declaración de XML desde el código PHP, tal como se indica a continuación:

```
<? echo '<?xml version="1.0" encoding="iso-8859-1"?>'; ?>
```

o también:

```
<?='<?xml version="1.0" encoding="iso-8859-1"?>'; ?>
```

Esta forma es la única solución que podemos emplear en aquellas situaciones donde no podamos modificar el fichero `php.ini`.

En el ejemplo de inclusión de ficheros con `require(fichero)` e `include(fichero)`, los ficheros incluidos tienen la extensión `.inc`. ¿Es lo más apropiado? No, es mejor que sea `.php`. ¿Por qué? Haz unas pruebas e intenta averiguar la razón [\(35\)](#).

Cuando trabajes con ficheros debes realizar todas las comprobaciones que puedas, ya que los ficheros suelen ser origen de muchos problemas durante el tiempo de ejecución de un programa: un fichero puede ser borrado, puede cambiar su nombre o puede ser movido de sitio. Además, otro problema son los permisos del sistema de archivos: el fichero puede existir, pero puede ser que no tengas permisos para acceder a él. Por tanto, **siempre que trabajes con ficheros, el primer paso debería ser comprobar que el fichero realmente existe y tienes permisos de acceso a él.**

Recuerda: en esta práctica todavía no debes usar sesiones, eso se verá en una próxima práctica.

El fichero de la página principal de un sitio web suele tener el nombre `index.html` cuando solo contiene código HTML e `index.php` cuando contiene código PHP. Sólo como curiosidad, los servidores web suelen tener configurada la siguiente lista de páginas por defecto para la página principal (36): `index.htm`, `index.html`, `index.php`, `index.php3`, `index.php5`, `index.php4`, `index.shtml`, `default.htm`, `default.html`, `index.py`, `default.shtml`, `index.pl`, `index.cgi` y `home.html`.

(1) <http://www.apachefriends.org/en/xampp-windows.html>

(2) <http://httpd.apache.org/docs/>

(3) <http://www.w3schools.com>

(4) <http://www.php.net/manual/es/>

(5) <http://www.addedbytes.com/cheat-sheets/php-cheat-sheet/>

(6) <http://notepad-plus.sourceforge.net/es/site.htm>

(7) <http://www.sublimetext.com/>

(8) http://www.c-point.com/php_editor.php

(9) *No es obligatorio que uses XAMPP, existen muchas otras alternativas como EasyPHP, WampServer o MAMP.*

(10) https://en.wikipedia.org/wiki/Clean_URL

(11) <https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador>

(12) http://httpd.apache.org/docs/2.2/mod/mod_alias.html

(13) <http://httpd.apache.org/docs/2.2/urlmapping.html>

(14) <http://httpd.apache.org/docs/2.2/mod/core.html#directory>

(15) *El servidor web Apache se puede configurar para que interprete como fichero PHP ficheros con otras extensiones.*

(16) <https://www.php.net/manual/es/language.variables.superglobals.php>

(17) <http://www.w3.org/Protocols/rfc2616/rfc2616.html>

(18) <https://www.php.net/manual/es/function.http-build-query.php>

(19) <https://codeigniter.com/>

(20) https://codeigniter.com/user_guide/general/welcome.html

(21) https://codeigniter.com/user_guide/overview/appflow.html

(22) https://codeigniter.com/user_guide/overview/mvc.html

(23) https://codeigniter.com/user_guide/general/urls.html

(24) https://codeigniter.com/user_guide/general/urls.html#removing-the-index-php-file

(25) https://en.wikipedia.org/wiki/Comparison_of_web_frameworks#PHP

(26) https://es.wikipedia.org/wiki/Marca_de_orden_de_bytes

(27) <https://www.w3.org/International/questions/qa-utf8-bom.es.php>

(28) <http://vibgyorLife.com/tech/article.aspx?catid=42>

(29) <http://www.apachefriends.org/en/faq-xampp-windows.html>

(30) *¿Qué puertos debo abrir para poder usar el Skype en Windows?:* <https://support.skype.com/es-es/faq/FA148/que-puertos-debo-abrir-para-poder-usar-el-skype-en-windows>

(31) <http://www.php.net/download-docs.php>

(32) *Si estás usando HTML5 para escribir las páginas web, todo lo que se explica a continuación no se aplica.*

(33) *No se aconseja utilizar esta sintaxis para escribir el código PHP porque puede ser que esté desactivada.*

(34) <http://php.net/manual/en/ini.core.php#ini.short-open-tag>

(35) En la página “What is .inc and why to use it?” tienes una buena explicación del problema:

<https://stackoverflow.com/questions/7129842/what-is-inc-and-why-to-use-it>

(36) El orden de configuración es importante porque el servidor web va a intentar encontrar los archivos en el orden indicado.