Práctica UD-4 Sesión-3

|  |  |  |  |
| --- | --- | --- | --- |
| Objetivos | * Aprender conceptos básicos de desarrollo web con PHP: estructura de archivos, sistemas de plantillas, MVC, composer. | | |
|
| Instrucciones de entrega | * Subid el proyecto o ficheros con el código que resuelve los ejercicios a Edmodo (no es imprescindible haber resuelto todos para entregar). | | |

## La función file\_get\_contents

* Esta función es similar a file(), excepto que file\_get\_contents() devuelve el fichero a un string, comenzando por el offset especificado hasta maxlen bytes. Si falla, file\_get\_contents() devolverá FALSE.
* file\_get\_contents() es la manera preferida de transmitir el contenido de un fichero a una cadena. Usa técnicas de mapeado de memoria, si está soportado por su sistema operativo, para mejorar el rendimiento.

Ejemplo

* Crea un nuevo proyecto en NetBeans (de tipo PHP->PHP Application) o en vuestro editor favorito con nombre “***PHPPracticaUD4Sesion3-file\_get\_contents***”.
* Copia los ficheros del repositorio *GitHub* en el proyecto y pruébalos:
  + <https://github.com/atomickarma/muobasicphp>
* Pruebe el resultado navegando a la URL:
  + <http://localhost/PHPPracticaUD4Sesion3-file_get_contents/index.php>

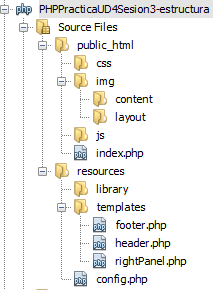
## 

## Estructura de un proyecto PHP

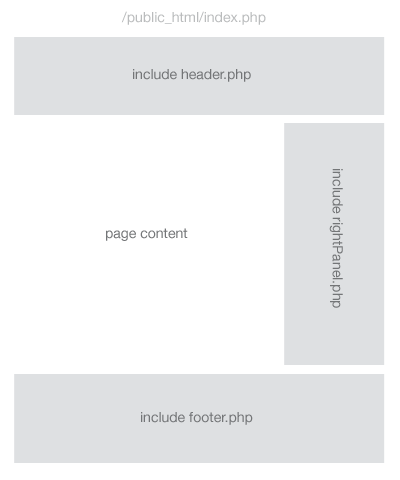
* PHP no define una estructura estándar de directorios para sus proyectos de forma que a lo largo del tiempo han aparecido diferentes opciones, veamos un ejemplo.

Ejemplo

* Crea un nuevo proyecto en NetBeans (de tipo PHP->PHP Application) o en vuestro editor favorito con nombre “***PHPPracticaUD4Sesion3-estructura***”.
* La estructura que queremos es la siguiente:



* El contenido de la página sería la siguiente:



* Crea un script php con nombre ***config.php*** con el siguiente contenido:

|  |
| --- |
| <?php  /\*  The important thing to realize is that the config file should be included in every  page of your project, or at least any page you want access to these settings.  This allows you to confidently use these settings throughout a project because  if something changes such as your database credentials, or a path to a specific resource,  you'll only need to update it here. \*/   $config = array(  "db" => array(  "db1" => array(  "dbname" => "database1",  "username" => "dbUser",  "password" => "pa$$",  "host" => "localhost"  ),  "db2" => array(  "dbname" => "database2",  "username" => "dbUser",  "password" => "pa$$",  "host" => "localhost"  )  ),  "urls" => array(  "baseUrl" => "http://example.com"  ),  "paths" => array(  "resources" => "/path/to/resources",  "images" => array(  "content" => $\_SERVER["DOCUMENT\_ROOT"] . "/images/content",  "layout" => $\_SERVER["DOCUMENT\_ROOT"] . "/images/layout"  )  ) );   /\*  I will usually place the following in a bootstrap file or some type of environment  setup file (code that is run at the start of every page request), but they work   just as well in your config file if it's in php (some alternatives to php are xml or ini files). \*/   /\*  Creating constants for heavily used paths makes things a lot easier.  ex. require\_once(LIBRARY\_PATH . "Paginator.php") \*/ defined("LIBRARY\_PATH")  or define("LIBRARY\_PATH", realpath(dirname(\_\_FILE\_\_) . '/library'));   defined("TEMPLATES\_PATH")  or define("TEMPLATES\_PATH", realpath(dirname(\_\_FILE\_\_) . '/templates'));   /\*  Error reporting. \*/ ini\_set("error\_reporting", "true"); error\_reporting(E\_ALL|E\_STRCT); |

* Crea un script php con nombre ***index.php*** con el siguiente contenido:

|  |
| --- |
| <?php // load up your config file require\_once(realpath(dirname(\_\_FILE\_\_) . "/../resources/config.php"));  require\_once(TEMPLATES\_PATH . "/header.php"); ?> <div id="container">  <div id="content">  <!-- content -->  </div>  <?php  require\_once(TEMPLATES\_PATH . "/rightPanel.php");  ?> </div> <?php require\_once(TEMPLATES\_PATH . "/footer.php"); |

* Crea un script php con nombre ***header.php*** con el siguiente contenido:

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>Simple Site</title>  </head>  <body>  <div id="header">  <h1>Simple Site</h1>  <ul class="nav global">  <li><a href="#">Home</a></li>  <li><a href="#">Articles</a></li>  <li><a href="#">Portfolio</a></li>  </ul>   </div> |

* Crea un script php con nombre ***rightPanel.php*** con el siguiente contenido:

|  |
| --- |
| <div id="siteControls">  <ul class="categories">  <li>PHP</li>  <li>HTML</li>  <li>CSS</li>  </ul>  <div class="ads">  <!-- ads code -->  </div>   </div> |

* Crea un script php con nombre ***footer.php*** con el siguiente contenido:

|  |
| --- |
| <div id="footer">  Footer content... </div> </body> </html> |

* Pruebe el resultado navegando a la URL:
  + <http://localhost/PHPPracticaUD4Sesion3-file_get_contents/index.php>
* Si quiere hacer más dinámico el contenido siga las instrucciones del tutorial:
  + <https://code.tutsplus.com/tutorials/organize-your-next-php-project-the-right-way--net-5873>
* Para finalizar mostramos la estructura que se recomienda actualmente y en la que se basan muchos framework:
* Directorio raíz del proyecto
  + ***bin/*** : ejecutables de línea de comandos
  + ***config/*** : archivos ejecutables
  + ***docs/*** : archivos de documentación
  + ***public***/ : archivos html o propios de la web
  + ***resources/*** : archivos js, css, img, etc
  + ***src/*** : código PHP
  + ***tests/*** : código de tests

## Sistemas/Motor de Plantillas (Templates)

* Una plantilla es un texto con referencias externas. Ejemplo:
  + *“Hola, {{nombre}}. Tu pedido {{referencia}}, realizado el día {{fecha}}, se realizó correspondiente. ".*
* Un motor de plantilla es una herramienta que reemplazará esas referencias externas (o cadenas de reemplazo), usando los datos que le proporciones.
* Existen multitud de motores de plantillas en PHP por ejemplo:
  + Smarty (independiente y muy popular)
  + Twig (de los creadores de Symfony)
  + Blade (el que usa Laravel)
  + Plates (nativo de PHP aunque hay que instalarlo aparte)

## Patrón MVC en PHP

* Vamos a hacer el siguiente tutorial desde el apartado patrón MVC al apartado carpeta public:
  + <https://www.jairogarciarincon.com/clase/programacion-orientada-a-objetos-en-php/patron-mvc>

## PHP con JS y Bootstrap

* En el siguiente tutorial se muestra cómo utilizar JS y Bootstrap en un proyecto simple de PHP:
  + <https://academy.leewayweb.com/un-dashboard-en-tiempo-real-basado-en-php-y-bootstrap/>

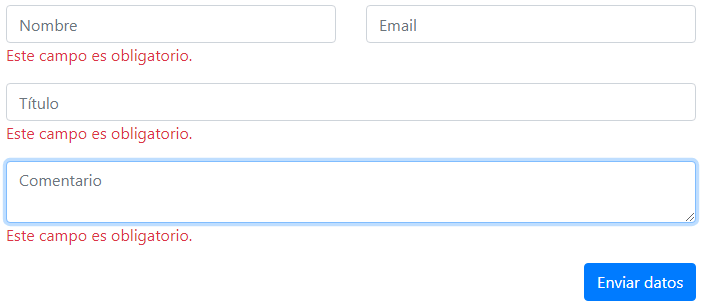
## PHP con Composer

* En el siguiente tutorial se muestra cómo utilizar Composer para un proyecto simple de PHP:
  + <https://www.frobiovox.com/posts/2016/08/16/basic-hello-world-with-composer-and-php.html>

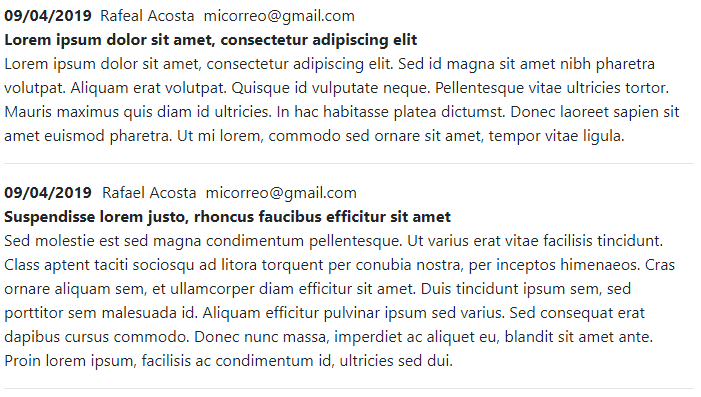
## Ejercicios

### Página de MicroBlogging

* Este ejercicio consiste en crear una página para la publicación de comentarios por parte de cualquier usuario. Constará de un formulario en su parte superior y a continuación los diferentes comentarios que los usuarios hayan ido añadiendo.
* Para su realización cree el proyecto ***“PHPPracticaUD4Sesion3-microblogging”*** que incluya un formulario como el que aparece en la imagen junto con el código PHP para su procesado:



* Debajo del formulario crea una tabla dinámica con una solo columna donde tengamos por cada fila el contenido de la forma: Fecha, Nombre, Correo, Título y Comentario, por ejemplo:



* Los comentarios no se deben perder en los sucesivos refrescos de la página.
* Una nueva sesión no mostrará ningún mensaje.

## Recursos

* <http://micmap.org/php-by-example/es/function/file_get_contents>
* <http://www.cellbiol.com/bioinformatics_web_development/chapter-4-adding-a-dynamic-layer-introducing-the-php-programming-language/dynamic-web-pages-with-php-a-simple-yet-useful-example/>
* <https://github.com/php-pds/skeleton>
* <https://www.makeuseof.com/tag/build-simple-php-website/>
* <https://stevesohcot.medium.com/sample-basic-php-template-for-file-structure-with-example-code-47ff6d610191>
* <https://code.tutsplus.com/es/tutorials/organize-your-next-php-project-the-right-way--net-5873>
* <https://desarrolloweb.com/articulos/plates-php-sistema-plantillas-templates.html>
* <https://www.victoriglesias.net/en-php-un-sistema-de-plantillas-es-prescindible/>
* <https://www.kabytes.com/programacion/7-sistemas-de-plantillas-templates-gratuitos-para-php/>
* <https://desarrolloweb.com/articulos/plates-php-sistema-plantillas-templates.html>
* <https://trasweb.net/snippets/construye-un-motor-de-plantillas-casero-en-php>