
theme : "white" transition: "zoom" highlightTheme: "darcula" customTheme : "lola_theme"

Tema 4: WordPress, Framework PHP

EI1042 - Tecnologías y Aplicaciones Web

EI1036- Tecnologías Web para los Sistemas de Información (2018/2019)

Professora: Dra. Dolores M^a Llidó Escrivá

[Universitat Jaume I.](#)

Índice

- Introducción Wordpress.
 - Conceptos de Wordpress.
 - Escritorio principal
 - WP CORE FILES.
 - Temas (Themes)
 - PHP
 - Etiquetas plantilla
 - Las etiquetas condicionales
 - Ganchos acción/Filtro.
 - Carga de la página principal en WP.
 - Plugins
 - Interacción C/S WP: CACHE
 - Proceso Autenticación
-

1. WordPress (WP)

Es un framework que permite gestionar bien:

- Un sitio web (<https://www.webempresa.com/>)
- Un Blog (<http://www.dulceida.com/>)
- Una combinación de ambos

Además de permitir añadir nuevas funcionalidades.

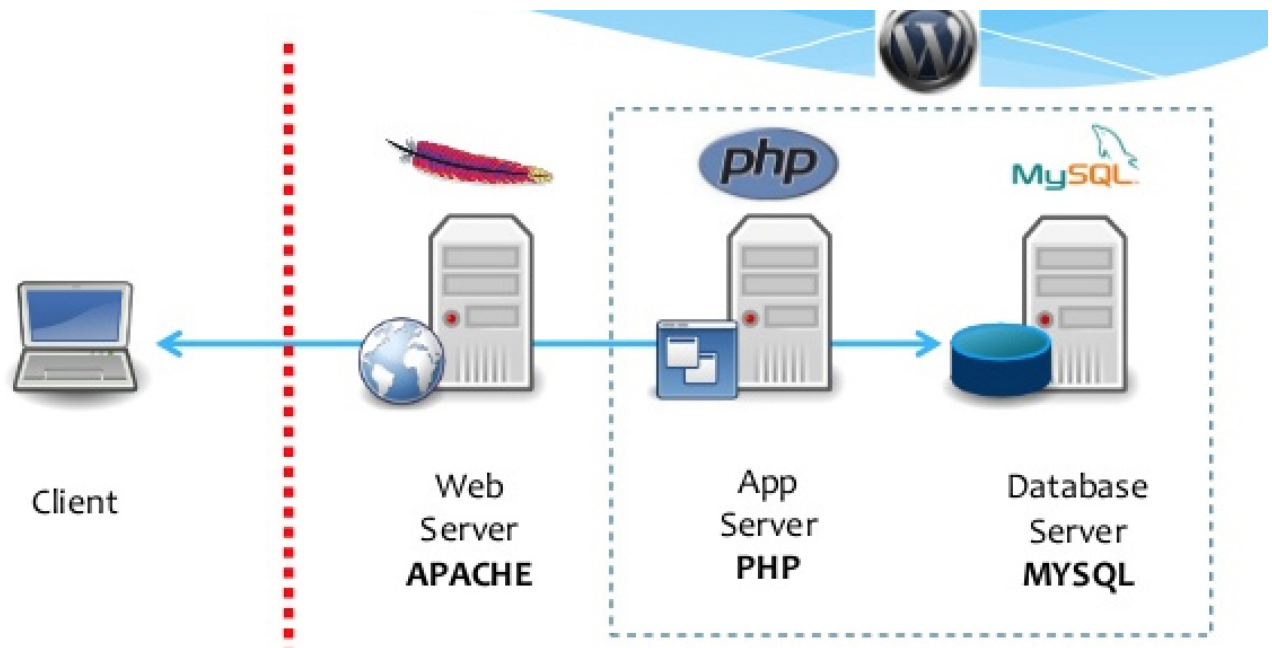
wordpress.org	wordpress.com
----------------------	----------------------

(software)

(hosting)

--

CMS WordPress



<https://www.slideshare.net/amikeliunas/introduction-to-wordpress-class-1>

--

Cuentas requeridas al instalar WordPress

- La cuenta del hosting: Sirve para acceder al hosting con FileZilla y para instalar WP.
- Cuenta en la base de datos de WP.
- Cuenta del super-administrador de tu instalación de WP.

2. Conceptos:

- Página
- Entrada (post)
- Comentario
- Categoría
- Etiqueta
- Atajos (shortcuts)
- Tema
- Plugin
- Widget
- Menu

Leer ↓

--

Lectura Individual

- **Página:** Contenido sin dependencia sitio web clásico.
- **Entrada o artículo:** la entrada (post) es la pieza básica de contenido de las bitácoras (blogs). La bitácora se compone de entradas con una determinada fecha de publicación.
- **Comentario:** publicaciones normalmente breves que se asocian a un artículo y que suelen estar hechas por terceras personas.
- **Categoría:** Las categorías son la forma de agrupar y clasificar contenido en WordPress. Admiten la construcción de jerarquías. Una página o una entrada puede pertenecer a una o varias categorías.
- **Etiqueta (tags)** son los conceptos que queremos señalar que aparecen en las páginas o artículos que publicamos. Las categorías son como una tabla de contenido, mientras la colección de etiquetas sería el glosario.
- **Atajos (shortcuts):** Consisten en un determinado texto entre las marcas [], que permiten añadir funcionalidad de algunos plugins en nuestra página web. Cuando la página es presentada al visitante, lo que hay entre las marcas se sustituye por el resultado de la función concreta que se ha invocado.

--

Conceptos:

- **Tema:** plantillas que se utilizan en WP para modificar la apariencia y diseño del sitio.
- **Plugin:** complementos de software que aumentan las capacidades y posibilidades de WP. Los plugins se usan para mejorar WP en diferentes áreas como marketing, redes sociales, seguridad, SEO, diseño web, contenido, tráfico web, etc.
- **Widgets:** pequeños bloques con herramientas que realizan funciones específicas y que están pensados para ubicarse en ciertas áreas del tema en curso, como el pie o la barra lateral (sidebar). Ayudan a darte un mayor control sobre el diseño y contenido de tu sitio web o blog. Los widgets se pueden expandir y usar de diferente forma dependiendo del tema y de los plugins que se instalen.
- **Menu:** Menus de navegación y acceso a distintos servicios del portal.

Cualquier widget o menú creado en WP se muestra, por defecto, en todas las páginas del sitio web.

--

Categorías / taxonomías

Las taxonomías de WP son un mecanismo de agrupamiento o categorización de los elementos de contenido y otros tipos de elementos de la aplicación.

- **Categorías:** actúan a modo de contenedores semánticos que sirven para organizar temáticamente las entradas del sitio web. Son jerárquicas, lo cual supone que pueden existir categorías, subcategorías, sub-subcategorías, etc.
- **Etiquetas:** funcionan a modo de descriptores semánticos del contenido de las entradas. No son jerárquicas (equivalente al glosario)
- **Categorías de enlaces:** permiten categorizar los enlaces, de forma semejante a como lo hacen las categorías de entradas. Desde la versión 3.5, tanto los enlaces como las categorías de enlaces ya no son

visibles en una instalación estándar de WP.

- Formatos de entradas: sirven para agrupar ciertos metadatos de información de las entradas, que pueden ser utilizados por un tema a fin de personalizar su presentación.

--

Uso taxonomias

- Las taxonomias contribuyen a jerarquizar el contenido y proporcionan “pistas” para que los buscadores puedan indexar los sitios web de forma eficiente.

URL amigables

- Página de entradas de la categoría correspondiente a la primera sesión del curso:
<https://cursoswp.educacion.navarra.es/cursowp2018/categoria/sesion-1/>. La “Sesión 1” es una categoría de entradas, que agrupa todos los elementos de contenido que se tratarán en dicha sesión.
- Página de las entradas que han sido marcadas con la etiqueta “widgets”:
<https://cursoswp.educacion.navarra.es/cursowp2018/etiqueta/widgets/>. La etiqueta “widgets” es un marcador semántico que sirve para poder agrupar todos los artículos que contengan dicho concepto.

--

¿Cuestiones?

3. Roles de Usuarios

- **SuperAdmin** – Quién cuenta con acceso a la característica de administración de la red de sitios completa. Una instalación de WP permite gestionar varios sitios en el mismo alojamiento.
 - **Administrador** – Quién tiene acceso a todas las características de administración de un sitio en particular.
 - **Editor** – Quién puede publicar y editar entradas, propias y de otros usuarios.
 - **Autor** – Quién puede publicar y editar sus propias entradas.
 - **Colaborador** – Quién puede escribir y editar sus propias entradas, pero no publicarlas.
 - **Subscriber** – Quién solamente puede editar su perfil.
-

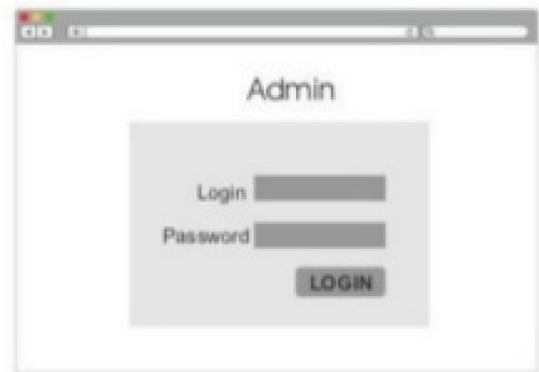
4. FrontEnd / Backend

Frontend



- Template System
- Displays Content






Backend




- Manage Content
- Install Extensions





Escritorio Principal (user: *Admin*)


 piruletas  4  0  Añadir


 Escritorio


Inicio
Actualizaciones 4


 Entradas


 Medios


 Páginas


 Comentarios


 WooCommerce


 Productos


 Apariencia


 Plugins

 Usuarios

 Herramientas

 All-in-One WP Migration

 Ajustes

 Cerrar menú

Escritorio

Did you know?


000webhost was created as an educational professional web hosting:

- WordPress sites hosted on hostinger
- SEO Optimization for WordPress - yo
- Daily backups, so your data will always
- Fast and dedicated support ready to
- Migration of your current WordPress
- Try Premium Hostinger offers from \$

TRANSFER
My site to Hostinger

Una nueva y moderna experi

Lleva tus palabras, medios y diseñ actualmente.



Panel de control

- Configurar el portal (tipo site/blog).
- Seleccionar tema.
- Gestionar usuarios.
- Gestionar taxonomías.
- Gestionar plugins.
- Actualizaciones.
- Gestionar los distintos recursos del site: multimedia (imagenes, videos...).

Leer ↓

--

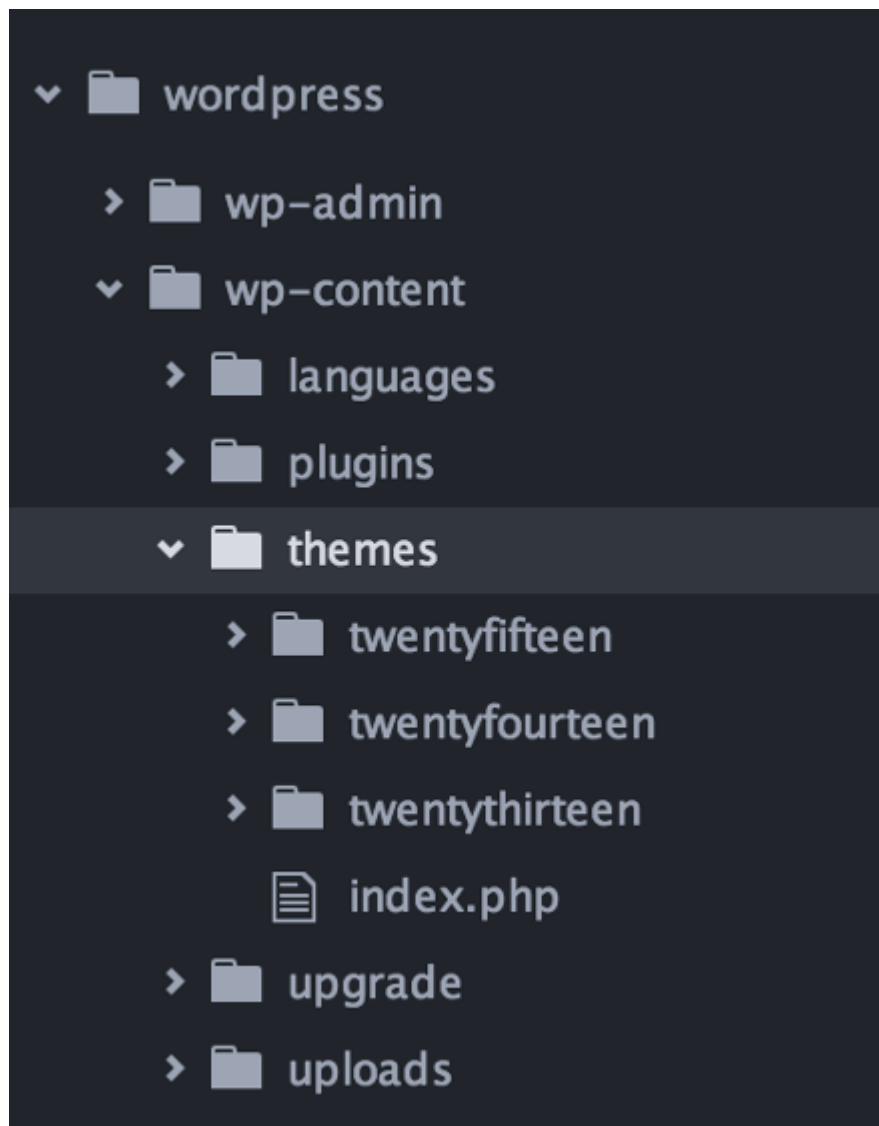
Tareas Administrador :

- Actualizaciones. En esta opción podrás ver todo lo relacionado con las diferentes actualizaciones disponibles para WP, incluyendo la versión de la plataforma, temas y plugins. Es importante mantener actualizado todo, por motivos de seguridad.
- Entradas. Desde aquí puedes administrarlas. También en esta sección podrás agregar y modificar tanto categorías como etiquetas.
- Medios. Agrega imágenes, archivos y hasta pequeños vídeos para usarlos en tus entradas o páginas.
- Páginas. A diferencia de las entradas, no tienen fecha de publicación.
- Comentarios. Área en donde administras todos los comentarios que se van publicando ya sea en tu blog o en tus páginas. Los puedes editar, eliminar o mandar al spam. También puedes seleccionar si prefieres primero verlos y aprobarlos o que se publiquen automáticamente.
- Información de tu tema o plantilla. Este apartado no aparece en todos los casos, más bien sólo cuando utilizas un tema "Premium" o que tiene su propio panel de opciones para modificar su estructura y diseño.

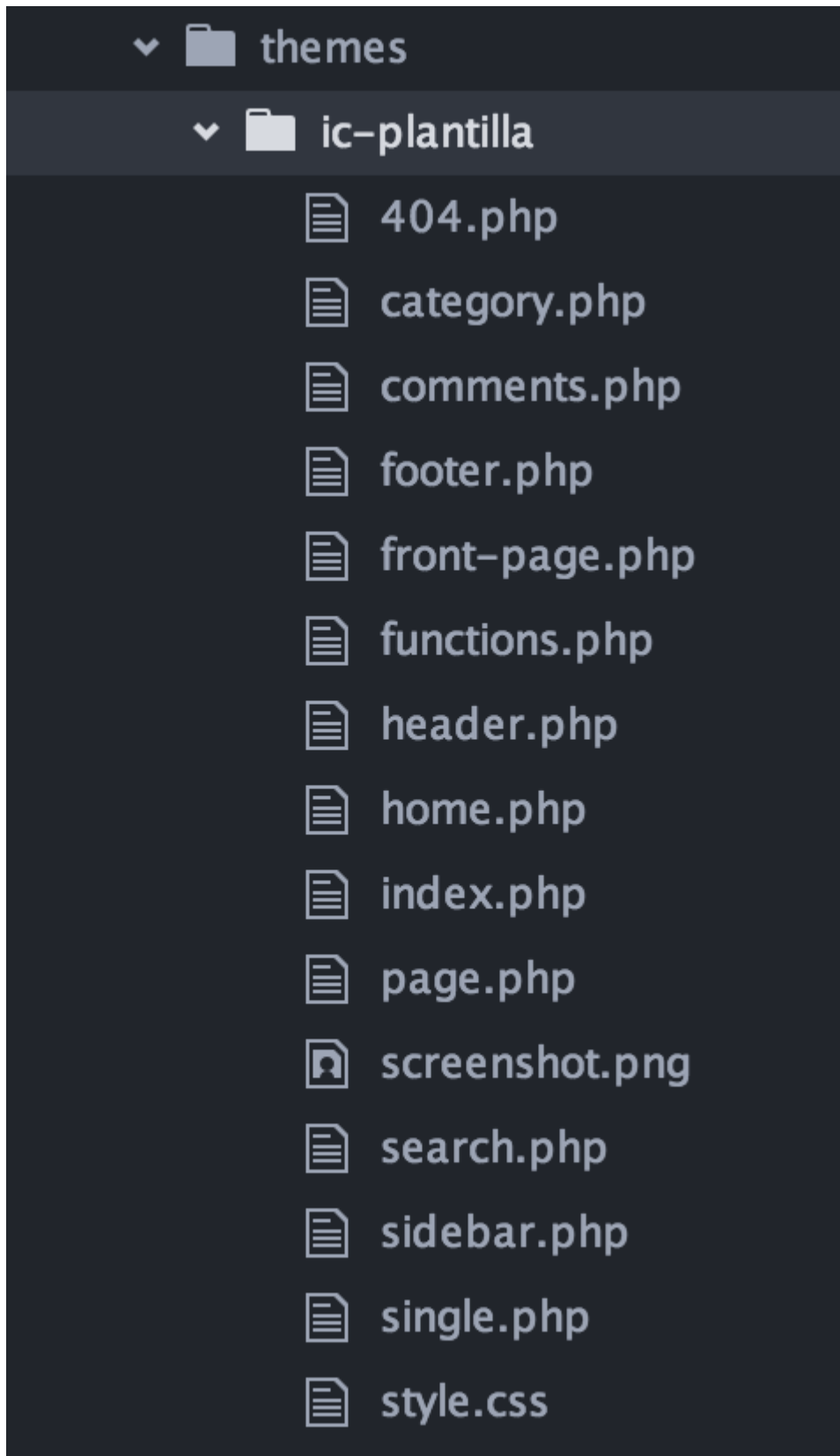
--

- Apariencia. En esta sección puedes agregar, instalar o eliminar nuevos temas, administrar los widgets, crear y modificar menús, editar el código de los archivos de tu tema (incluyendo el CSS) y también personalizar el título y la portada estática de tu tema (la página de inicio).
- Plugins. Agrega, modifica, instala o elimina nuevos plugins.
- Usuarios. Aquí puedes modificar tu perfil, agregar tus links de redes sociales, cambiar tu alias, tu email e información básica. Si otras personas van a editar tu sitio o publicar contenido, en esta sección puedes crearles un perfil y definir su rol.
- Herramientas. Importar y exportar son las opciones disponibles inicialmente. Si instalas ciertos plugins, su panel de opciones podría aparecer en esta sección.
- Ajustes. Toda la configuración básica de tu sitio se hace en esta parte. También podrás modificar las opciones de cualquier plugin que instales en esta sección. De entrada tendrás acceso a configurar las opciones generales, de escritura, de lectura, de comentarios, de medios y de enlaces permanentes.

5. Sistem de Ficheros WP



6. Anatomía de un tema



HomePage: index.php (blog) o front-page.php-> home.php -> page.php(static web)

--

Muy Interesante : <https://yoast.com/wordpress-theme-anatomy/>

functions.php

functions.php es donde agrega funcionalidad personalizada a su tema:

- Hojas de estilo CSS (incluyendo style.css).
- Archivos JavaScript.
- Registro de áreas del menú de navegación y áreas de widgets.
- Filtrado el contenido de la página según el usuario

Leer ↓

--

Listado Ficheros Template

- index.php: Plantilla página de inicio del site. **OBLIGATORIO**.
- style.css: fichero CSS principal.
- header.php: Plantilla cabecera común. Todo lo que contiene la etiqueta o el menú de navegación principal.
- footer.php: Plantilla del pie de página (por ejemplo el copyright o los enlaces a la información legal).
- sidebar.php : Plantilla de una barra lateral (sobre todo cuando se trata de un blog) o con widgets (por ejemplo un buscador, un listado de categorías o los posts más visitados).
- front-page.php: Plantilla de la página de inicio del blog. Está especialmente pensada para que sea un listado de entradas, es decir, la portada de un blog.
- home.php : Plantilla por defecto de inicio si se ha seleccionado que la página de inicio es **página estática**, o sea un portal web.

--

- single.php :Plantilla que muestra una entrada completa por defecto.
- page.php: mostrará por defecto cualquier página que creemos.
- category.php : Sirve para mostrar un listado de posts de una categoría específica.
- comments.php : Es la plantilla a la que llamaremos dentro de single.php para poder añadir los comentarios a nuestros posts.
- search.php: Esta plantilla nos permite realizar búsquedas en el site.
- 404.php: Plantilla que se mostrará cuando un enlace esté roto o no funcione.
- functions.php : Permite crear zonas de menú y de widgets, así como personalizar algunos parámetros que WordPress trae por defecto.

CHILD THEMES

- Los temas secundarios (child themes) son la forma en que los diseñadores y desarrolladores pueden realizar pequeños ajustes a páginas específicas de un sitio sin tener que crear un tema completo para ellos.
- El tema hijo se pone en el directorio Temas con el nombre del tema padre seguido de -child.
- Debe tener 2 ficheros: index.php (copiar el del padre) y style.css

Ejemplo: twentyseventeen-child:

--

Ejemplo style.css

```

Theme URI:      http://example.com/twenty-seventeen-child/
Description:    Twenty Seventeen Child Theme
Author:        John Doe
Author URI:     http://example.com
Template:       twentyseventeen
Version:       1.0.2
License:       GNU General Public License v2 or later
License URI:   http://www.gnu.org/licenses/gpl-2.0.html
Tags:          one-column, two-columns, right-sidebar, flexible-header,
accessibility-ready, custom-colors, custom-header, custom-menu, custom-
logo, editor-style, featured-images, footer-widgets, post-formats, rtl-
language-support, sticky-post, theme-options, threaded-comments,
translation-ready
Text Domain:   twenty-seventeen-child
*/

/* =Aquí empieza la personalización de tu tema
----- */

```

- Revisar la referencia a la plantilla padre en "template"

-- Según la jerarquía de WP:

- Primero se busca el fichero en el tema hijo.
- Luego se busca en el tema por defecto.

The WordPress template hierarchy

WP busca los archivos de plantilla en el siguiente orden:

- **Plantilla de página**: : plantilla personalizada asignada, WordPress busca ese archivo y, si lo encuentra, lo usa.
- **page-{slug}.php**:: plantilla especializada que contiene el slug (babosa) de la página, o sea el nombre del recurso.
- **page-{id}.php**:: plantilla especializada que incluye el id de la página.
- **page.php**:: plantilla de página predeterminada del tema.
- **singular.php**:: plantilla para una sola publicación.
- **index.php**:: archivo de índice del tema para representar páginas.

--

Ampliar ↓

<https://wp hierarchy.com/>



--

Ejemplo: Pagina del usuario de los autores, "david", tenga una identificación numérica de 3.

¿Existe el archivo `author-david.php` en el tema secundario? No...

¿Existe el archivo `author-david.php` en el tema principal? No...

¿Existe el archivo `author-3.php` en el tema secundario? No...

¿Existe el archivo `author-3.php` en el tema principal? No...

¿Existe el archivo `author.php` en el tema secundario? ¡Sí!

8. PHP

¿Como se añaden funciones propias en WP?

- Creando en las plantillas las funciones: **Code Snippets**.
- Creando un **plugin**.
- Añadir código personalizado a WordPress (en **functions.php**). Para que no se borren al actualizar los plugins directamente en la carpeta `/mu-plugins/`. ("Must Use" plugins).

9. PHP: Etiquetas de plantilla (tags)

Son funciones PHP para incluir fácilmente archivos de plantilla desde el tema en otro archivo o para mostrar información de la base de datos.

Etiquetas de plantilla para cargar plantillas:

WP	PHP
get_header()	include('header.php')
get_sidebar()	include('sidebar.php')
get_footer()	include('footer.php')
get_search_form()	include('searchform.php')

Ventaja: podemos personalizar que código cargando bien la plantilla por defecto u otra personalizada.

Cuestiones:

Problema: ¿Como cargar la cabecera del fichero `header-custom.php` en un tema?

Solución `<?php get_header('custom'); ?>`

Cargará el fichero header-custom.php en vez de header.php

Problema: Mostrar el pie de página en la página principal, pero no en ninguna otra página Solución: agregue `get_footer()` al final de `index.php`, pero no en `page.php`. ¿Por que?

--

Etiquetas de plantilla para mostrar información de la base de datos:

- `bloginfo()` - muestra el nombre de su sitio web según se define en el Panel de control del administrador
- `single_post_title()` – muestra el título de la publicación vista actualmente cuando se utiliza en `single.php`
- `the_author()` – muestra el autor del mensaje visto en ese momento
- `the_content()` – muestra el texto principal de una publicación o página
- `the_excerpt()` – el extracto de la publicación o página

--

Las etiquetas condicionales

“etiquetas condicionales” o conditional tags, permiten determinar en qué circunstancias debe producirse un determinado comportamiento de la aplicación.

- `is_home()`
- `is_front_page()`
- `is_single('excursion-a-piedramillera')` Que algo ocurra solo si el visitante está viendo la entrada denominada “Excursión a Piedramillera”; la etiqueta que permite conseguir este comportamiento es
- `in_category('sesion-1')` Que algo ocurra si el visitante está leyendo una entrada de la categoría “Sesión 1”. Para tal fin, hay que utilizar la siguiente etiqueta:

--

```
<?php
if ( is_admin() ) {
    // we are in admin mode
    require_once( dirname( __FILE__ ) . '/admin/plugin-name-admin.php' );
}
```

Sintaxis alternativa de estructuras de control ¶

Condigo html oculto

```
<?php if ( is_user_logged_in() ) {
    echo "CUIDADO estas Logueado ok?";
} else {
    echo "CUIDADO NO Estas logeado.";
}?>
```

PHP ofrece sintaxis alternativa para : if, while, for, foreach, y switch.

En cada caso, la forma básica de la sintaxis alternativa es cambiar la llave de apertura por dos puntos 😊 y la llave de cierre por endif;, endwhile;, endfor;, endforeach;, o endswitch;, respectivamente.

--

Solución:

```
<?php if ( is_user_logged_in() ): ?>
    <h1 >CUIDADO Estas logeado ok? </h1>
<?php else: ?>
    <h1 >CUIDADO NO Estas logeado. </h1>
<?php endif; ?>
```

10. Ganchos

Los Plugins de WordPress interactúan con código del núcleo de WP utilizando ganchos. Hay dos tipos diferentes de ganchos.

- Ganchos de acción (para agregar / quitar funciones).
- Ganchos de filtro (Para modificar datos producidos por funciones).

```
add_action( 'user_register', 'crf_user_register' );
add_filter( 'autenticarClientes', 'myplugin_auth_signon', 30, 3 );
```

--

Ganchos de acción (hooks)

Los hooks de acción o acciones son disparadas cuando pasa algo, como que se cargue la página, se inicie una sesión...

- Los ganchos de acción son funciones que tienen generalmente funciones asociadas a ellas.
- Algunos ganchos existen para su uso por plugins, no tienen funciones conectadas de forma predeterminada.
- Los ganchos de acción permiten que los complementos conecten sus propias funciones y que se ejecuten en varios puntos de la carga de una página.

--

Ejemplo Hook predefinido:

```
<head>
<meta charset="<?php bloginfo( 'charset' ); ?>">

<?php wp_head(); ?>
</head>````
```

– En la plantilla de encabezado de su tema, se invoca el gancho de acción `wp_head()` para incluir el encabezado HTML predeterminado de WordPress.

– `wp_head()` es un gancho de acción predefinido en WordPress.

Plugin API/Action Reference.
https://codex.wordpress.org/Plugin_API/Action_Reference

--

Definir función action hook

Para asociar nuestras propias funciones a ganchos de acción, debemos usar `add_action()`:

```
add_action( 'compass_in_footer', 'compass_smallprint', 20 );````
```

** definimos que nuestra función `compass_smallprint`, se ejecute cada vez que se llame a la acción `compass_in_footer`, con una prioridad de 20.

* Para llamar a la acción se utiliza `do_action()`

```
do_action( 'compass_in_footer' );````
```

--

Asociar funciones a un Action Hook

* Se puede asociar a una acción más de una función, de forma que se ejecutará primero la de más prioridad. Si no se indica la prioridad por

defecto es 10.

```
```php
add_action('compass_in_footer', 'compass_colophon');
add_action('compass_in_footer', 'compass_smallprint', 20);
do_action('compass_in_footer');
```

¿que función se ejecuta antes en el ejemplo anterior? ¿Por que no llamar directamente a las funciones?

--

¿Por que no llamar directamente a las funciones?

- Puedes enlazar más de una funcion de la misma accion,
- Puedes fijar la prioridad para que ellos se ejecuten en el orden que los quieres.

--

## Borrado funciones action hook

Podemos borrar nuestras funciones con `remove_action( 'compass_in_footer', 'compass_smallprint', 20 );`

O todas las funciones con: `remove_all_actions( 'compass_in_footer' );`

## Ganchos de filtro

Útil si tienes opciones en tu tema o plugin que quieras sobrescribir una opcion por default, o si estas creando un tema padre que puede tener elementos sobrescritos de un tema hijo.

- Una función de filtro te permite modificar los datos resultantes que son devueltos por funciones ya existentes y deben estar enganchados a los ganchos de filtro.
- Al igual que los ganchos de acción se llaman en varios puntos del guión y son contextuales.
- Adición de filtros mediante `add_filter()`
- Ejecución del gancho de filtro con `apply_filters()`.
- Los filtros se pueden poner en functions.php.

## apply\_filters

`apply_filters( string $tag, mixed $value )`

- Es la función que se encarga de ejecutar los ganchos.
- tiene tres parametros : el nombre del hook de filtro, el valor que quieres filtrar(ejem. el valor por default), y variables opcionales las cuales pasaraas a las funciones enlazadas al filtro.
- Una lista completa de los ganchos de filtro: [https://codex.wordpress.org/Plugin\\_API/Filter\\_Reference](https://codex.wordpress.org/Plugin_API/Filter_Reference)

Los filtros predefinidos no hace falta definir nosotros la funcion de apply\_filters, directamente lo ejecutamos.



Por ejemplo:

```
apply_filters('the_content', string $content)
ya definida en:

wp-includes/post-template.php
```

--

## add\_filter()

```
add_filter(string $tag, callable $function_to_add, nt $priority = 10, int
$accepted_args = 1)
```

- Función que permite añadir funciones de filtro:
- es un callback con 4 parámetros:
  - \$tag puede ser cualquier gancho de WP.
  - \$function\_to\_add la función del callback que se ejecuta al activar el gancho.
  - \$arg2 Prioridad.
  - \$arg3 Número de argumentos de la \$function\_to\_add

--

### Ejemplo filter hook :

Para filtrar el contenido de cualquier post quitando dobles espacios

```
add_filter("the_content", "mfp_Fix_Text_Spacing");

// Automatically correct double spaces from any post
function mfp_Fix_Text_Spacing($the_Post)
{
 $the_New_Post = str_replace(" ", " ", $the_Post);
 return $the_New_Post;
}
apply_filters('the_content', string $content)
```

--

Lectura:<https://code.tutsplus.com/es/articles/wordpress-actions-and-filters-whats-the-difference--cms-25700>

Ejemplo complejo:

- <https://getflywheel.com/wp-content/uploads/2015/06/Anatomy-of-a-WordPress-Theme.png>
- <https://piruletas.000webhostapp.com/Participa/>

---

¿Como genera la página html en WP?

- index.php carga gancho en wp-blog-header.php el cual se encarga de inicializar todo el wordpress y si está inicializado el Template carga su CSS y plantillas.
- llamando al frontpage (blog)/ home.php(portal)
- y estos cargarán todas las otras plantillas de las distintas secciones con las distintas etiquetas de plantilla. (get\_header(),get\_sidebar() |get\_footer())
- Si el usuario está autenticados además carga una cabecera con el menú del rol del usuario.

- 
- ¿Quien genera la página principal/home?
  - ¿Y las noticias/entradas?
  - ¿Y la página Participa?
  - ¿Y en el plugin woocommerce como se genera la tienda?
- 

## 11 Plugins

---

WordPress sólo tiene instalado el plugin Akismet, cuyo principal objetivo es evitar cualquier ataque de SPAM en el sitio web en cuestión.

Sistema ficheros de un plugin.

```
/plugin-name
 plugin-name.php
 uninstall.php
 my-plugin.php
 index.php
 readme.md
 /languages
 /includes
 /admin
 /js
 /css
 /images
 /public
 /js
 /css
 /images
```

**Ampliar ↓**

--

Directorios

- admin, para los php con funcionalidad de back-end
- css, para almacenar allí nuestras hojas de estilos
- includes, para archivos php auxiliares
- js, para almacenar nuestros scripts js
- languages, para la internacionalización de nuestro plugin

- public, para los php con funcionalidad de front-end

--

Ficheros:

- index.php, (Obligatorio)
- my-plugin.php, que es el fichero principal de nuestro plugin, y
- uninstall.php que se ejecutará cuando un usuario borre nuestro plugin para realizar acciones de limpieza en BBDD.
- README.md
- CHANGELOG.md

--

Lectura Individual <https://www.cssigniter.com/use-custom-theme-plugin-wordpress-site-customizations/>

- ¿Conclusiones?
- ¿Cuándo es conveniente utilizar un plugin o un child-theme?

---

## 12 Widget

---

- Los widgets, son bloques en los que puedes añadir una gran selección de elementos: iconos de redes sociales, listado de entradas más vistas en el blog, categorías del sitio web, contacto...
- Para crear un widget simplemente se crea un fichero php en la carpeta /wp-content/plugins/my-widget/ el cual debe contener la instrucción `register_widget('name')` lo cual registra en el WP un widget llamado 'name'. Que podemos insertar en cualquier parte de la página web seleccionándolo de la lista de widgets activos.

--

### Ejemplo de añadir un widget¿?

(<https://www.axarnet.es/blog/como-crear-un-widget-en-wordpress/>)

*Mi\_Widget.php*

```
<?php.
/*
Plugin Name: Mi Widget
Plugin URI: http://wordpress.org/extend/plugins/#
Description: Este es un ejemplo de cómo crear un Widget
Author: "Tu nombre"
Version: 1.0
Author URI: http://ejemplo.com/
*/
add_action('widgets_init', function(){
```

```
register_widget('Mi_Widget');
});
```

## 13. CustomFields para las entradas

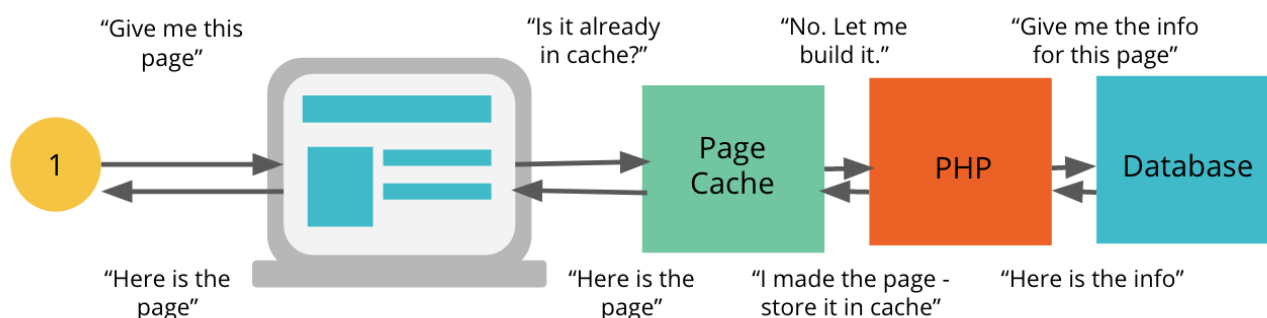
- Permite a los autores asignar campos personalizados a una entrada. Esta información arbitraria es conocida como metadatos.
- Los metadatos se manejan en pares de clave/valor.

Los autores pueden asignar estos e incluso crear nuevos.

Después de escribir la entrada del blog, ir al área titulada Custom Fields para modificar o asignar "customFields".

## 14. Interacción C/S WP: Cache

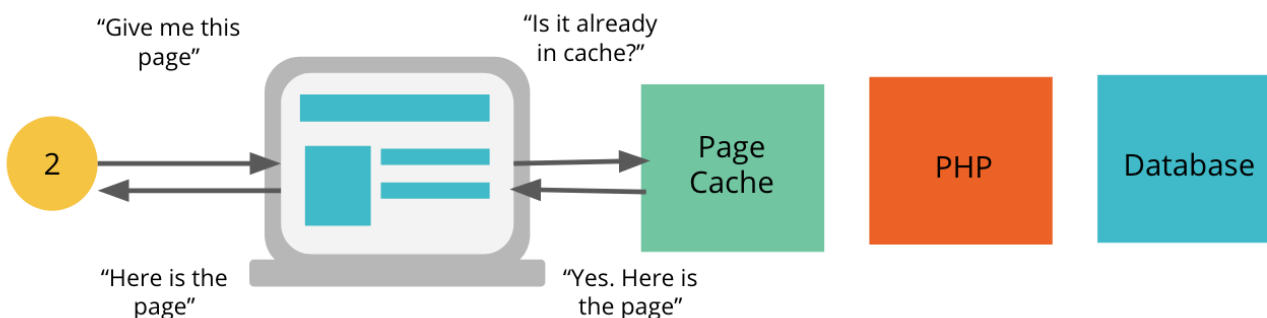
<https://wpengine.com/support/cookies-and-php-sessions/>



¿que pasa con las sesiones la 2 vez? ¿Y con las cookies?

--

No hay interacción con PHP.



JavaScript permite visualizar las cookies.

¿Y el objeto SESSION PHP?

--

## SESIONES

---

- Por defecto, WordPress no le dará la posibilidad de recuperar datos de los usuarios para mejorar su experiencia de usuario.
- El propio WordPress no retiene sesiones, pero si algunos plugins o temas una vez que probablemente se activan.
- WordPress no utiliza sesiones PHP nativas. En cambio, depende en gran medida de las cookies para la autenticación y almacena cualquier información adicional sobre una sesión autenticada en la base de datos.

--

Lectura individual en casa

Uso sesiones:

<https://slicejack.com/building-custom-wordpress-login-page/>

Ejemplo Login <https://pressjitsu.com/blog/wordpress-sessions-performance/>

---

## 14. WEB: Autenticación de usuarios

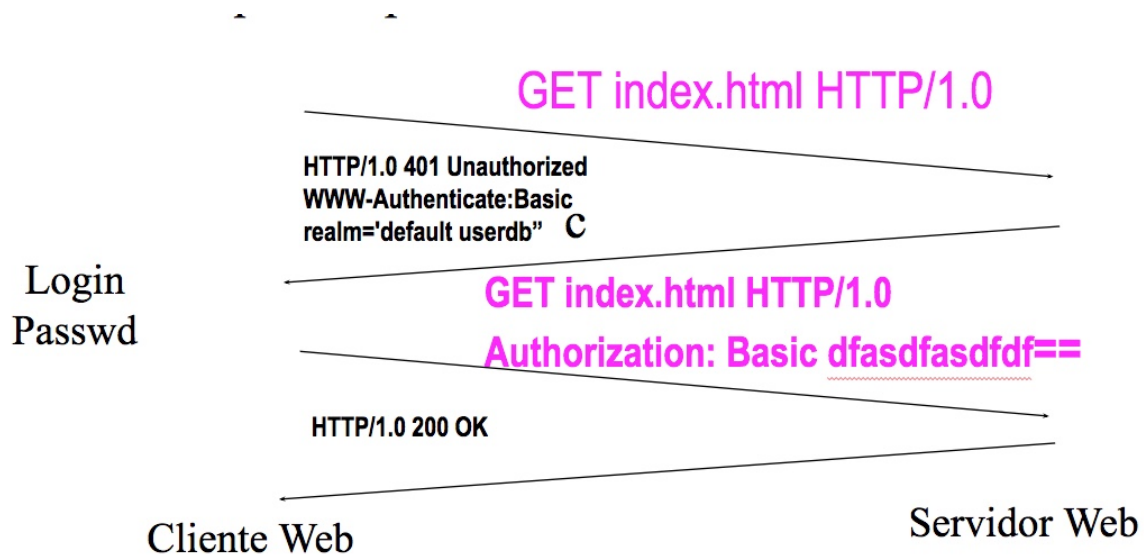
---

- Una cuestión frecuente en un sitio web es controlar el acceso de los usuarios a una zona determinada del mismo (autorización), para ello se requiere solicitar generalmente la Autenticación previamente.
- Autenticación requiere credenciales o pruebas de identidad.
- La autenticación de usuarios puede realizarse:
  - Autenticación en el Servidor: En Apache los ficheros **.htacacces**.
  - Autenticación en el Cliente: Firma Digital.
  - Autenticación por Programa: Escribir un programa para controlar el acceso de los usuarios.

--

### Proceso autenticación básica en Servidor:

Autenticación básica: Solicita al cliente un usuario y contraseña, que viajan encriptadas con codificación base 64 bits



--

## Configuración Autenticación Básica

- Configurar fichero .htaccess en directorio que se requiera.

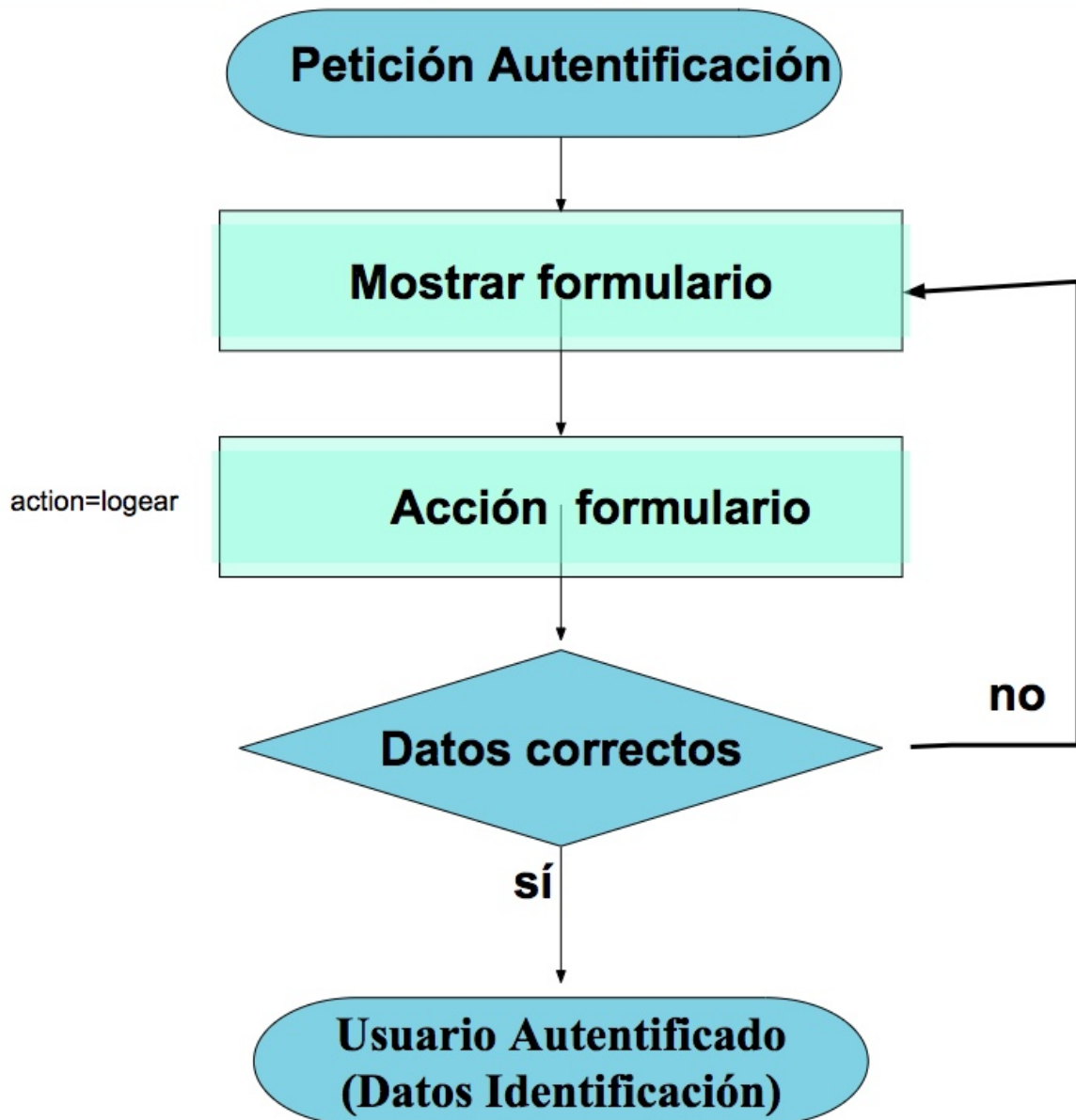
```
#./htaccess
AuthType Basic
AuthName "Password Required"
AuthUserFile /home/alxxx/privado/.htpasswd
Require valid-user
```

- Ejemplo: <https://piruletas.000webhostapp.com/teoria/T4/Auth/session.php>

```
#Generar fichero password en linux
>htpasswd -c /home/alxxx/privado/.htpasswd usuario
```

--

## Autenticación Básica por Programa



--

<https://piruletas.000webhostapp.com/teoria/T4/basicAuth.php>

```
<?php
$valid_passwords = array ("Pepe" => "Catala","lola"=>"lola");
$valid_users = array_keys($valid_passwords);
$user = $_SERVER['PHP_AUTH_USER'];
$pass = $_SERVER['PHP_AUTH_PW'];
$validated = (in_array($user, $valid_users)) && ($pass ==
$valid_passwords[$user]);

if (!$validated) {
header('WWW-Authenticate: Basic realm="My Realm"');

header('HTTP/1.0 401 Unauthorized');

die ("Not authorized");
```

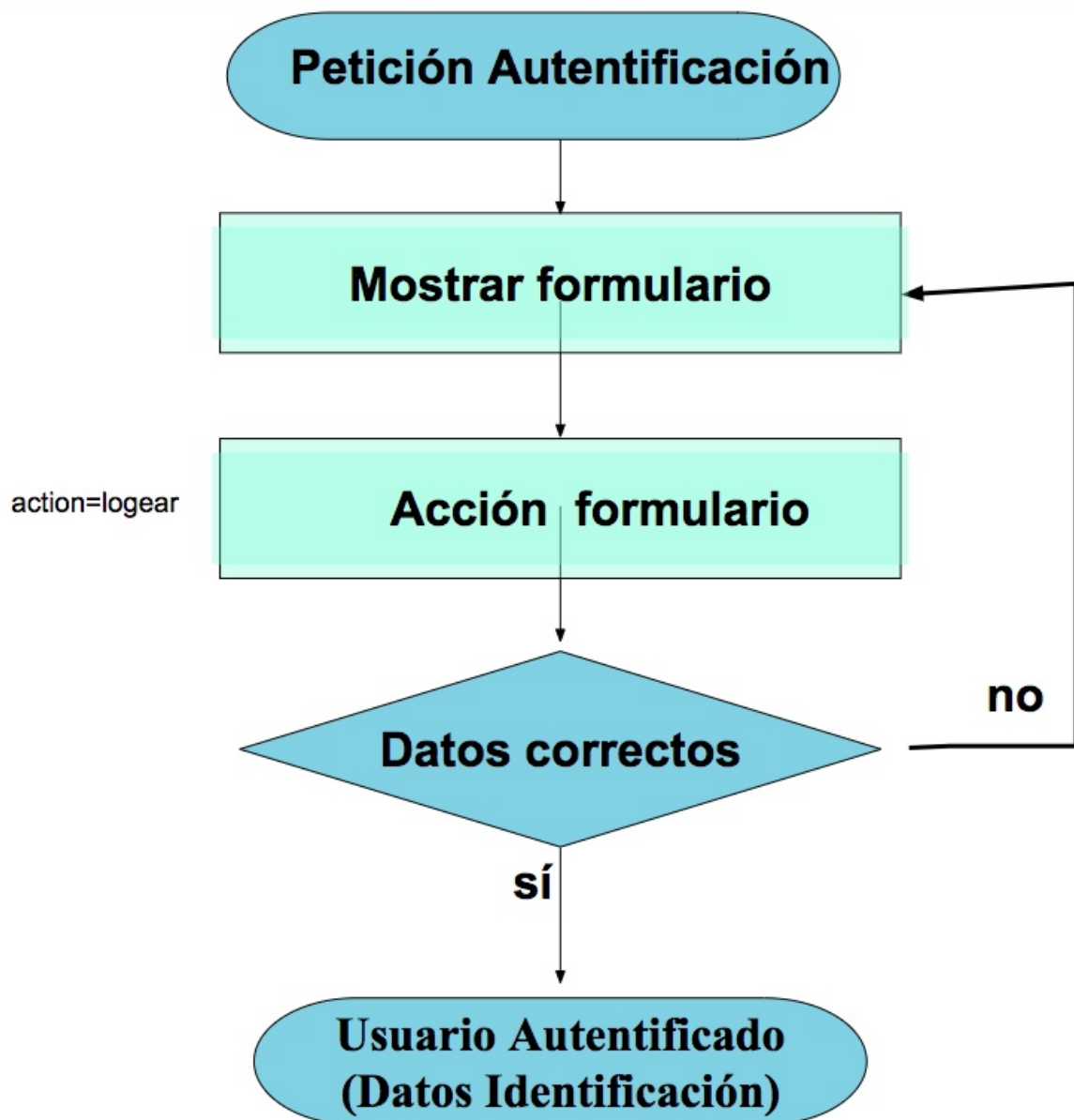
```
}
// If arrives here, is a valid user.

echo "<p>Welcome $user.</p>";

echo "<p>Congratulation, you are into the system.</p>";

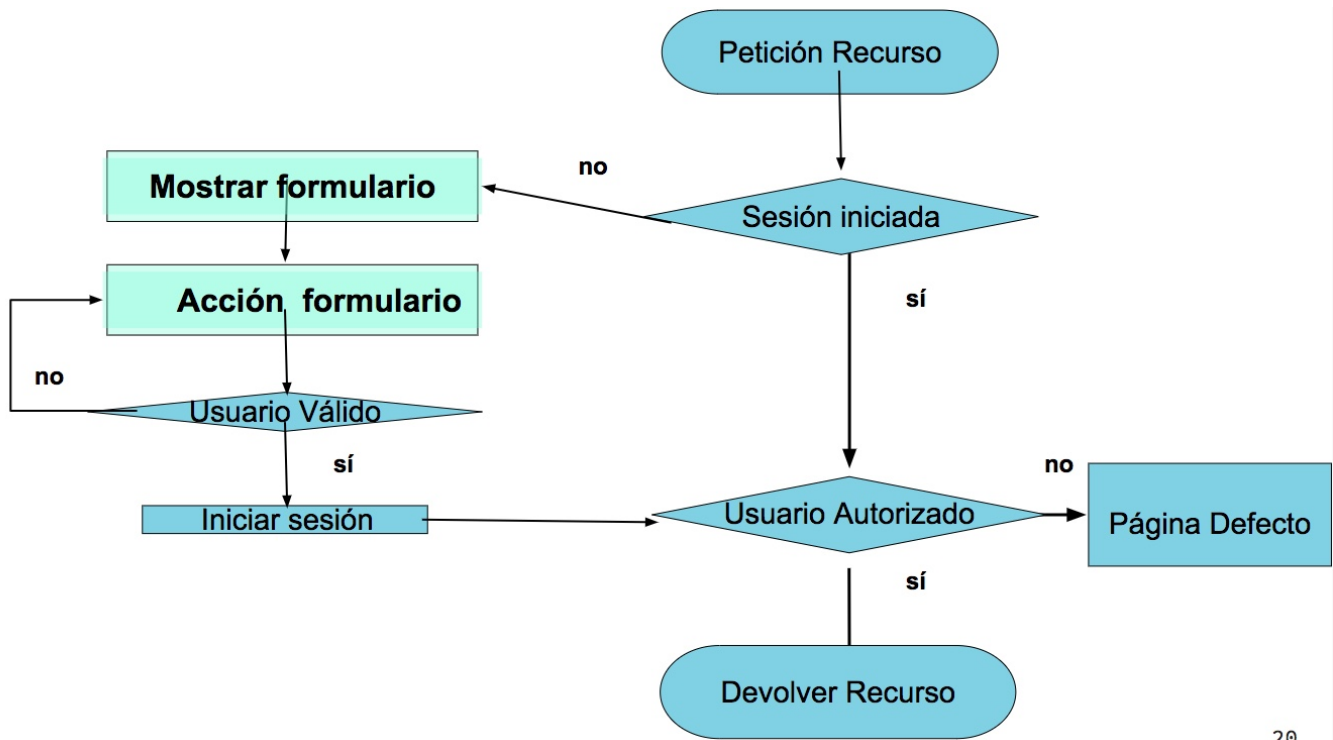
?>
```

--



Autorizar recursos a los usuarios





20

## Cuestiones:

- ¿Hay que pedir autorización a todos los recursos que se soliciten?
- ¿La acción “redirect” es propia de php o del protocolo http?
- ¿Qué entiendes por usuario cliente, visitante,

gestor, administrador?

- ¿Que podríamos hacer al cambiar el rol del usuario?

## WP y Gestión de usuarios:

WP como tiene un sistema de autenticación tiene por defecto un sistema de login que almacena en la BD información sobre el usuario.

Por tanto tiene:

- una variable global para acceder al BD: \$wpdb
- Un objeto wp-user que nos permite gestionar los datos de los usuarios y comprobar que la autenticación es correcta.

## Objeto wp\_user:

Parametros wp\_user ([https://developer.wordpress.org/reference/classes/wp\\_user/](https://developer.wordpress.org/reference/classes/wp_user/))

- ID (int): identificador usuario.
- roles (array): los roles del user.
- first\_name (string): nombre usuario.
- last\_name (string): apellido usuario.

Ejemplo para comprobar que el usuario esta registrado

```
if (!$user = $wpdb->get_row($wpdb->prepare(
 "SELECT * FROM $wpdb->users WHERE $db_field = %s", $value
)))
```

---

## Autenticación WP: wp\_authenticate\_cookie

---

Hook para comprobar si un usuario está autenticado:

- wp\_authenticate\_cookie( WP\_User|WP\_Error|null \$user, string \$username, string \$password )

devuelve: (WP\_User|WP\_Error) WP\_User on success, WP\_Error on failure.

[https://developer.wordpress.org/reference/functions/wp\\_authenticate\\_cookie/](https://developer.wordpress.org/reference/functions/wp_authenticate_cookie/)

---

## Bibliografía

<https://visual.ly/community/infographic/computers/wordpress-theme-anatomy>

---

## ¿Dudas?

