MEDAC
Instituto Oficial de Formación Profesional

DESPLIEGUE DE APLICACIONES WEB
TÉCNICO EN DESARROLLO DE APLICACIONES WEB

Documentación de aplicaciones

14

ÍNDICE

 ISBN: 978-84-18983-26-9

OBJETIVOS

Descubrir para qué sirve documentar.

Diferenciar los diferentes tipos de documentación.

Conocer los diversos formatos que podemos utilizar.

Aprender qué son los sistemas de documentación colaborativa.

Usar generadores de documentación.

/ 1. Introducción y contextualización práctica

En esta unidad, hablaremos de la importancia de documentar las aplicaciones que desarrollamos. Aprenderemos a diferenciar entre el tipo de documentación que elaboramos para un cliente y la que hacemos para nosotros mismos.

También, veremos los diferentes formatos que podremos usar para entregar la documentación y aprenderemos qué son las herramientas colaborativas.

Para terminar, explicaremos en qué consisten los generadores automáticos de documentación y como utilizarlos.

Planteamiento del caso práctico inicial

A continuación, vamos a plantear un caso a través del cual podremos aproximarnos de forma práctica a la teoría de este tema.

Escucha el siguiente audio donde planteamos la contextualización práctica de este tema, encontrarás su resolución en el apartado Resumen y resolución del caso práctico.

Fig. 1. Documentación de aplicaciones.

Audio intro. "Cambios en una aplicación"
https://bit.ly/36wtpew

/ 2. Documentación de aplicaciones

A la hora de crear una aplicación, es importante dejar constancia del funcionamiento y de todo aquello que sea relevante para el proyecto. **Ser riguroso en la documentación de los procedimientos y protocolos facilitará la resolución de errores y ayudará a que todos los miembros del departamento sean capaces de entender cómo funciona,** por ejemplo, una función PHP, una clase de Java o cualquier sentencia de cualquier lenguaje que usemos en nuestros desarrollos.

Un ejemplo de documentación de una aplicación son los comentarios en los archivos que incluyen el código fuente. Como ya sabemos, estos comentarios serán ignorados por el compilador y nos darán la posibilidad de dejar reflejada información que facilitará la lectura de dicho código.

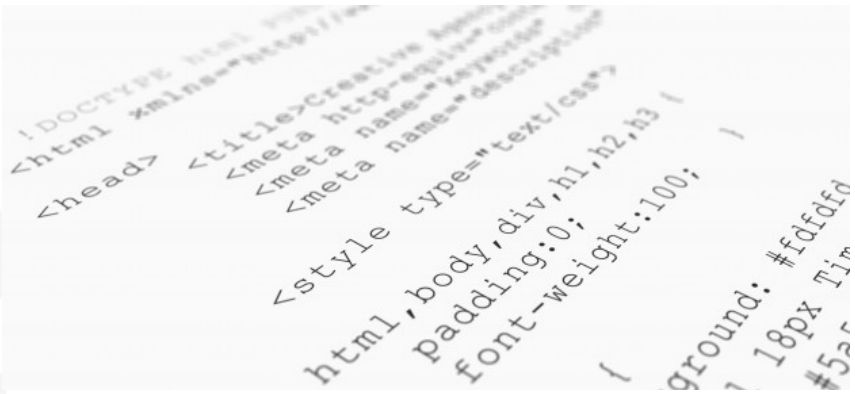

Fig. 2. Ejemplo de código HTML.

Aportar todos los datos relevantes es una necesidad que nace en el mismo momento en el que surge el primer boceto del proyecto. Desde ese instante, **registraremos las ideas más importantes y todo lo que necesitemos que no se olvide.** Pero esta documentación no acaba con el desarrollo del proyecto, también será necesario **documentar todos aquellos problemas que puedan surgir durante la vida útil de la aplicación.**

El proceso de documentación debe especificar aspectos tales como la **finalidad** de la aplicación, los **motivos** que llevaron a desarrollar dicha aplicación y el **funcionamiento** de la misma.

De manera global, podemos clasificar la documentación generada en dos grupos: documentación para uso externo y documentación de uso interno.

- **Documentación para uso externo:** estos documentos serán los que entregaremos al **usuario final**, ya sea en forma de manual de utilización o como un resumen de las especificaciones técnicas necesarias para instalar y usar la aplicación.

- **Documentación para uso interno:** estos documentos serán utilizados por el **propio equipo o departamento** que esté diseñando o realizando el mantenimiento de la aplicación.

  En estos documentos, debe aparecer cualquier dato necesario para poder realizar cambios o corregir errores, así como cualquier otra información útil.

/ 3. Formatos

Otro aspecto importante al elaborar la documentación es el formato que elegiremos para presentar la información. Por ejemplo, en el mundo internet, es muy común que una aplicación web tenga un apartado de **FAQ** (*Frequently Asked Questions*) en formato HTML, que no deja de ser una manera de resolver dudas y explicar el funcionamiento de la *app*.

Entre los diferentes formatos que podemos utilizar, están:

- **HTML:** es uno de los formatos más utilizados, ya que al combinarlo con CSS, nos permitirá crear una guía de uso visualmente atractiva, quedando integrada con el diseño general de la aplicación.

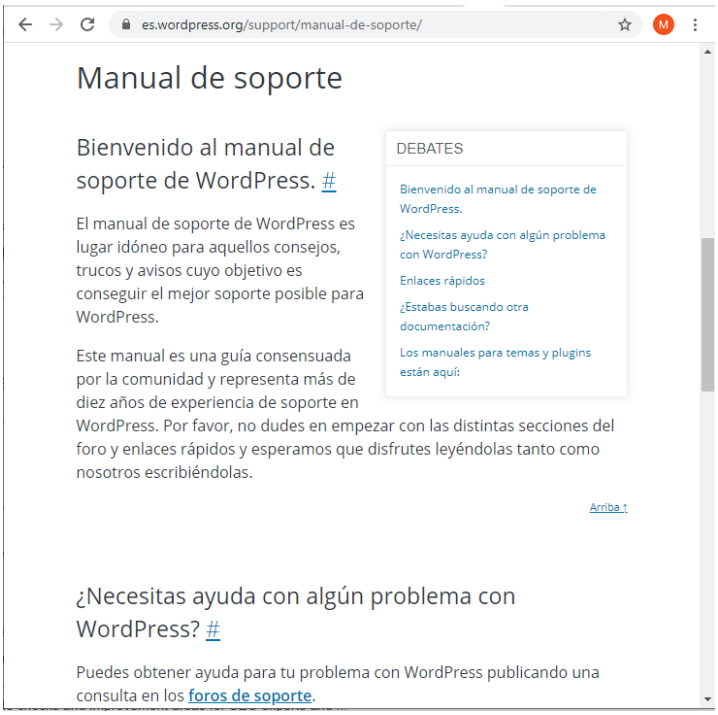

Fig. 3. Manual de soporte de Wordpress. *https://es.wordpress.org/support/manual-de-soporte/*

- **PDF:** el formato PDF nos permitirá realizar una documentación en formato tradicional, incluyendo un índice con la tabla de contenidos. A diferencia del formato HTML, un documento PDF no se podrá modificar de forma directa, por lo tanto, HTML es más versátil respecto a posibles cambios o modificaciones.

- **TXT:** este formato solo permite el uso de caracteres de texto, es decir, no será posible incluir imágenes o hipervínculos.

Cabe destacar que para guardar la **documentación de uso externo**, se podrá utilizar el mismo **servidor que aloje la aplicación.** De esta manera, estará siempre accesible para el consumidor final. No obstante, la **documentación interna** suele ser de tipo confidencial o privada, y necesitaremos un **sitio seguro para clasificar y almacenar los documentos.**

Para cubrir esa necesidad, se puede utilizar NAS (disco duro en red), un servidor de archivos SMB o cualquier otro dispositivo que nos permita tener la información segura, estableciendo un sistema de control de accesos.

Audio 1. "El servidor para gestión documental"
https://bit.ly/3lwAfoS

/ 4. Comentarios en Java-Javadoc

En programación, existen una serie de normas que definen las buenas prácticas a la hora de escribir código. Estas normas o **estándares de programación** sirven, principalmente, para conseguir una uniformidad en los desarrollos. Además, estos estándares indican la estructura básica que debería tener el código escrito para hacer más fácil la comprensión y conseguir un mejor rendimiento.

Sin embargo, cada desarrollador tiene su propio estilo y no siempre se seguirán al pie de la letra. El uso de comentarios en el código nos ofrece otra manera de facilitar la comprensión y algunos entornos de desarrollo incluyen funciones para automatizar el uso de estos comentarios.

En este apartado, vamos a estudiar la utilidad Javadoc, que Wikipedia define siguiente manera: "Javadoc es una utilidad de Oracle para la generación de documentación de APIs en formato HTML a partir de código fuente Java. Javadoc es el estándar de la industria para documentar clases de Java. La mayoría de los IDEs los generan automáticamente".

Entre los IDE que utilizan Javadoc para generar los comentarios, podemos encontrar Eclipse. Partiendo de la ventana principal, podremos ejecutar el **asistente de generación de Javadoc** pinchando en el botón **proyecto > Generar Javadoc,** tal y como se muestra en la siguiente imagen:

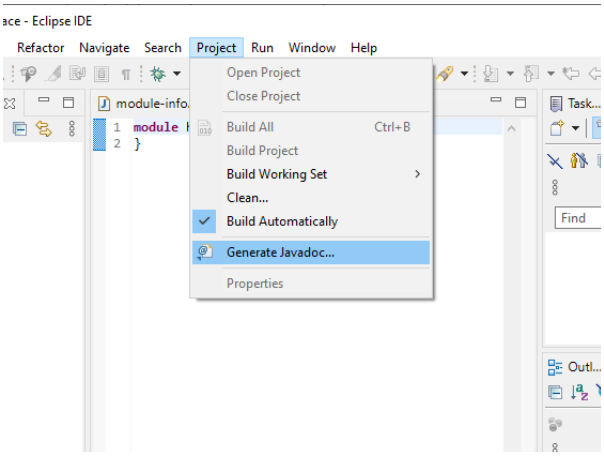

Fig. 4. Generación de Javadoc en Eclipse.

A continuación, podemos ver cómo sería un ejemplo de comentario con Javadoc.

```
/**
* Connects and logins to the FTP server.
*
* @throws FTPConnectionException If an I/O error occurs.
* @throws FTPLoginException If the login operation did not succeed.
*/
public void login() throws FTPConnectionException, FTPLoginException {

  // method body

 }
```

Código 1. Ejemplo de comentario Javadoc. https://www.codejava.net/ides/eclipse/how-to-generate-javadoc-in-eclipse

Vídeo 1. "Eclipse en Javadoc"
https://bit.ly/36tKvK1

/ 5. Caso práctico 1: "Documentación para clientes"

Planteamiento: Leticia trabaja desarrollando aplicaciones webs para pymes y le han encargado realizar la documentación relativa a una aplicación de gestión de personal.

Entre las tareas pendientes está generar los manuales de uso, notas técnicas para la instalación y configuración, y documentación relativa al funcionamiento interno de la aplicación.Cada uno de los documentos generados tendrá diferentes objetivos y, por lo tanto, debe elaborarse teniendo en cuenta los conocimientos que tendrán las diferentes personas que lo puedan leer.

Nudo: ¿Qué tipos de documentación tendrá que generar? ¿Qué formatos deberá utilizar para cada uno de los documentos?

Desenlace: Teniendo en cuenta los diferentes perfiles que accederán a la documentación, Leticia tendrá que discriminar entre documentos de uso interno y documentos de uso externo. Los manuales de uso que entregará al cliente se consideran de uso externo y no contendrán información sensible. Del mismo modo, de las notas con los requisitos de instalación y configuración se podrán generar dos versiones: una versión de uso externo para entregar al cliente y otra de uso interno con datos ampliados. Estos documentos se pueden entregar al cliente en formato PDF.

Fig. 5. Diferentes tipos de documentación para diferentes usuarios.

La documentación referente al funcionamiento de la aplicación debe ser confidencial y de uso interno. En estos documentos, se dejará constancia de cualquier dato que pueda ser necesario para resolver problemas futuros o realizar modificaciones. Se pueden utilizar formatos digitales, como PDF o HTML, y, además, es interesante complementar con comentarios en el código fuente.

/ 6. Herramientas colaborativas

La documentación de las aplicaciones se convierte en un aspecto imprescindible en grandes equipos de trabajo donde cada línea debe ser entendible por todos los miembros sin necesidad de analizar el programa completo.

En el proceso de documentación interna, los comentarios dentro del código pueden no ser lo suficientemente efectivos como para analizar y clasificar todos los detalles. Además, estos comentarios limitan el acceso a la información al personal cualificado. Por este motivo, se recurre a **aplicaciones externas que ofrecen la capacidad de almacenar datos de manera sencilla.**

Un sistema práctico y funcional es, por ejemplo, una wiki. Una wiki permite crear una base de datos, categorizando la información mediante el uso de etiquetas. Un ejemplo de wiki es la Wikipedia, que almacena más de 6 millones de artículos.

Fig. 6. Recorte de https://en.wikipedia.org/wiki/Main_Page

Habitualmente, en una wiki, podemos analizar tres capas bien definidas por cada una de las páginas del sistema:

- **Código HTML** que se mostrará al responder peticiones de clientes. El servidor generará este código a partir del código fuente.
- **Plantillas** utilizadas para definir la posición de los elementos mostrados en una página.
- **Código fuente** que se almacena en el servidor, habitualmente, en texto plano y visible solo al editar artículos.

Existe un gran número de wikis de código abierto que podremos utilizar en el proceso de documentación. Algunas de ellos son:

- **MediaWiki:** plataforma de código abierto escrita en PHP y desarrollada por Wikipedia.
- **TikiWiki:** aplicación *open source* y multiplataforma que permite colaborar a la hora de crear o editar documentación. Desarrollada sobre PHP.
- **XWiki:** esta herramienta está escrita en Java y, al igual que las anteriores, permite crear portales para la gestión colaborativa.

Además de las wikis, podemos utilizar otros sistemas específicos para la documentación de código. Es el caso, entre otros, de PhpDocumentor.

/ 7. PhpDocumentor

PhpDocumentor aparece en el año 2000 como una adaptación de Javadoc, permitiendo generar documentación directamente a partir del código fuente PHP. Está escrito en PHP y se distribuye con licencia *open source* LGPL.

Entre las principales características de PhpDocumentor, podemos destacar las siguientes:

- Permite exportar a formatos HTML, PDF y CHML.
- Estandarización del formato de la documentación.
- Genera un índice para facilitar la búsqueda de contenidos.
- Uso de plantillas.
- Sistema de seguimiento de errores.

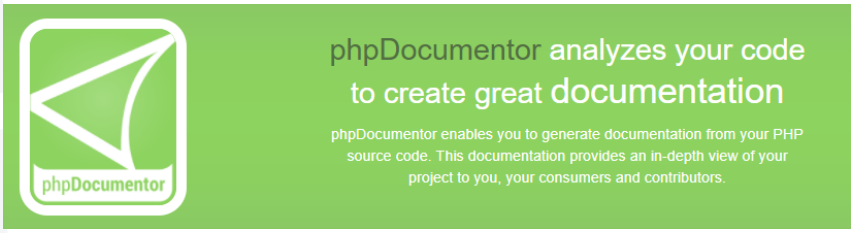

Fig. 7. Captura de la web de PhpDocumentor. https://www.phpdoc.org/

En el siguiente enlace, podremos encontrar toda la documentación del proyecto: https://docs.phpdoc.org/latest/

Vídeo 2. "PhpDocumentos hamarc en puesta e instalación"
https://bit.ly/3mB3d8k

a) Instalación PhpDocumentor

Como ya sabemos, PhpDocumentor está creado en PHP, por lo que necesitaremos un servidor web para poder realizar la instalación. En la web oficial de PhpDocumentor, encontraremos los requisitos y dependencias que necesitaremos cumplir.

La instalación de PhpDocumentor la realizaremos utilizando el comando **pear.** En el siguiente extracto de código, veremos el proceso de instalación de *php-pear*, la configuración del directorio de trabajo en /var/www y, por último, la creación de la carpeta donde PhpDocumentor podrá almacenar los ficheros procesados.

```
sudo apt-get install php-pear
pear config-set data_dir /var/www
sudo pear config-set data_dir /var/www
sudo pear install --alldeps phpDocumentor
sudo mkdir /var/www/docs
sudo chown www-data /var/www/docs/
```

Código 2. Instalación de PhpDocumentor en Ubuntu 18.04.

A continuación, podríamos ejecutar el comando **phpdoc** o acceder a PhpDocumentor a través de un navegador web:

Fig. 8. Interfaz web de PhpDocumentor. https://www.phpdoc.org/

7.1. DocBlock

Un bloque de documentación o **DocBlock** es el elemento más básico que podemos encontrar en PhpDocumentor. El formato de un DocBlock es el siguiente:

```
/**
*
*/
```

Código 3. Ejemplo de DocBlock.

Un DocBlock está compuesto por tres secciones que seguirán siempre el mismo orden:

- Descripción corta
- Descripción larga
- Etiquetas

```
<php
public function isLoggedIn();
 /**
 * Devuelve la información del usuario sobre la cuenta
 *
 * This method is used to retrieve the account corresponding
 * to a given login. <b>Note:</b> it is not required that
 * the user be currently logged in.
 *
 * @access public
 * @param string $user user name of the account
 * @return Account
 */
 public function getAccount($user = '');
}
?>
```

Código 4. Ejemplo de DocBlock. http://www.juntadeandalucia.es/servicios/madeja/sites/default/files/historico/1.4.0/contenido-libro-pautas-103.html#Generar_documentacion_separada_para_diferentes_tipos_de_usuario. -

Habría que destacar que el uso de **plantillas** en las herramientas de documentación nos aportará un estilo estandarizado que facilitará la comprensión del código y la reducción de información redundante.

Las plantillas son otro tipo de DocBlock, y se diferencian de uno básico por su cabecera, que debe comenzar por /**#@+

```
/**#@+
 *
 */
```

Código 5. Ejemplo de plantilla de DocBlock.

Al utilizar plantillas, no será necesario documentar el tipo de variable cada vez que la utilicemos.

```
 <?php
class WisdomDispenser
{
 /**#@+
 * @access protected
 * @var string
 */
 private $firstSaying = 'Obey the golden rule.';
 private $secondSaying = 'Get in or get out.';
 private $thirdSaying = 'Everything is relative';
 /**#@-*/
}
 ?>
```

Código 6. Ejemplo de plantilla de DocBlock.http://www.juntadeandalucia.es/servicios/madeja/sites/default/files/historico/1.4.0/contenido-libro-pautas-103.html#Generar_documentacion_separada_para_diferentes_tipos_de_usuario

En este enlace, encontrarás información ampliada del uso de DocBlocks y plantillas.

/ 8. Otros asistentes de documentación

Además de PhpDocumentor, podemos encontrar una gran variedad de programas que asisten en la labor de documentar una aplicación web. De entre todas ellas, destacaremos **Doxygen y Natural Docs.**

a) Doxygen

Doxygen es un *software* que genera documentación a partir del código fuente. Este programa está escrito en C++ y, a diferencia de PhpDocumentor, que solo soporta PHP, admite otros lenguajes como Java, C o C++.

Fig. 9. Doxygen Logo.

Al estar escrito en C++, no necesita un servidor web para funcionar, y en la web oficial, encontraremos los instaladores y archivos binarios para Windows, Mac y Linux.

En el siguiente enlace, podremos ampliar información sobre Doxygen:

http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/110

b) natural Docs

Natural Docs se presenta como un generador de documentación todoterreno, siendo capaz de trabajar con 21 lenguajes de programación. Este *software* está desarrollado en Perl y utiliza licencia de *software* libre GNU.

Natural Docs es multiplataforma y permite su ejecución en sistemas Windows, Mac y Linux. Como punto negativo y a diferencia de PhpDocumentor y Doxygen, diremos que Natural Docs solo puede exportar el resultado en formato HTML.

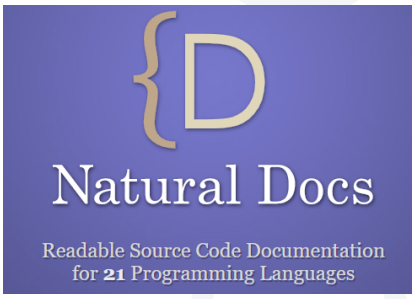

Fig. 10. Logo Natural Docs.

En el sitio web oficial de Natural Docs, podremos encontrar información actualizada y documentación:

https://www.naturaldocs.org/

/ 9. Caso práctico 2: "Generador de comentarios de aplicación web"

Planteamiento: En la empresa en la que trabaja Román, se están planteando utilizar un sistema que les facilite generar documentación del código fuente que desarrollan.

Principalmente, se dedican a desarrollar aplicaciones en Java o PHP, aunque también realizan módulos escritos en C++.

El principal objetivo sería encontrar una aplicación que permita documentar cualquier aplicación, con independencia del lenguaje de programación que utilicen.

Nudo: ¿Qué sistema se adapta a sus necesidades?

Desenlace: Román deberá utilizar un generador que le permita documentar varios lenguajes. Por este motivo, JavaDoc, que solo es capaz de procesar código Java, quedaría descartado. Lo mismo ocurre con PhpDocumentor, ya que solo podrá generar comentarios a partir de programas escritos en PHP.

Doxygen podría ser una buena opción por compatibilidad con los lenguajes requeridos por la empresa.

Del mismo modo, Natural Docs podría ser otra buena opción por su amplia compatibilidad con más de 20 lenguajes de programación.

Fig. 11. Los generadores de comentarios permiten que cualquier componente del equipo de desarrollo pueda comprender el código.

/ 10. Resumen y resolución del caso práctico de la unidad

En esta unidad, hemos aprendido para qué sirve y cómo realizar correctamente la **documentación** de aplicaciones webs. Asimsimo, hemos explicado los **diferentes tipos de documentación** que podemos generar dependiendo de su finalidad y los diferentes formatos que podemos utilizar.

Por otro lado, hemos podido conocer qué son las **herramientas colaborativas** que permitirán tener una base de conocimiento del proyecto con una interfaz web.

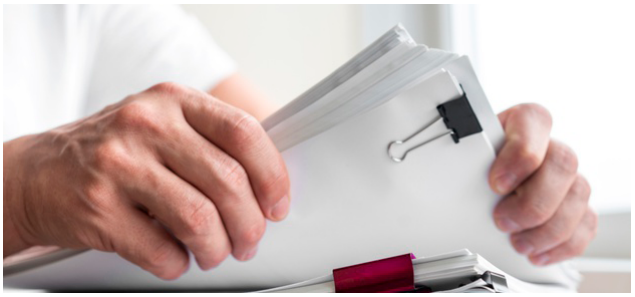

Fig. 12. El servicio de directorio como sistema de autenticación centralizado.

Por último, hemos hablado de los **generadores automáticos de comentarios** y de su funcionamiento. Hemos prestado una especial atención a los **DockBlocks** y a las plantillas, que son elementos necesarios para estandarizar y evitar información redundante.

Resolución del caso práctico inicial

En el caso práctico inicial, se plantea la siguiente situación: en una empresa de desarrollo web, se ha incorporado un nuevo trabajador y necesita hacer cambios en una aplicación que lleva varios años en producción. Para poder realizar las modificaciones, necesita familiarizarse con el código y conocer todas las funciones y variables.

Esto supone una inversión de tiempo bastante elevada, y al tener otras tareas pendientes que se van acumulando, es una situación delicada.

Si se hubiera documentado correctamente, el proceso de desarrollo, así como las funciones y variables mediante alguno de los procesos estudiados, Ainara podría solucionar el problema en un tiempo mucho menor, puesto que ahora tendrá que trabajar en un sentido totalmente opuesto al natural, intentando, primero, descifrar qué función realiza cada parte del código, así como cada uno de los elementos de los que puede constar la aplicación, y llevarlo a un documento ordenado.

/ 11. Bibliografía

Cabello, A.L.C. (2020). Implantación de aplicaciones web en entornos internet, intranet y extranet. ifcd0210 - desarrollo de aplicaciones con tecnologías web. IC Editorial

/ 12. Webgrafía

https://zonaphp.com/generadores-de-documentacion/
http://www.juntadeandalucia.es/servicios/madeja/sites/default/files/historico/1.4.0/contenido-libro-pautas-103.html
https://w3.ual.es/~rguirado/posi/1-3_Documentacion.pdf
https://codigofacilito.com/articulos/guia_codigo
https://es.wikipedia.org/wiki/Wiki
https://info.tiki.org/HomePage
https://www.xwiki.org/xwiki/bin/view/Main/WebHome
https://en.wikipedia.org/wiki/List_of_wiki_software
http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/255
https://pear.php.net/package/PhpDocumentor/docs/latest/phpDocumentor/tutorial_phpDocumentor.howto.pkg.html#intro
doxygen.nl/index.html
https://www.naturaldocs.org/
http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/110