# Tema 6: Documentación y control de versiones.

"La programación es una carrera entre ingenieros de software luchando para construir programas cada vez más grandes, mejores y a prueba de idiotas, y el universo intentando producir cada vez más grandes y mejores idiotas. Por ahora, gana el universo. " Rich Cook.

## 1.- Documentación de aplicaciones web.

¿Qué conviene documentar en una aplicación? Tres aspectos fundamentales de la aplicación:

* **La interfaz:** qué hace (no como lo hace) una función o un método de una clase, qué parámetros hay que pasar y qué devuelve.
* **La implementación:** indicar cómo está implementada cada función, cómo se lleva a cabo cada paso, por qué se utiliza determinada variable, qué algoritmo se utiliza, qué hacen los métodos privados de una clase.
* **La toma de decisiones:** por qué se ha implementado de determinada forma y no de otra la aplicación.

Normalmente la información sobre la implementación no necesita salir del código pero, por el contrario, la información de la interfaz conviene pasarla a un documento indenpendiente del código fuente (manual de uso).

La persona que necesite utilizar una determinada librería de clases o funciones tendrá toda la información necesaria: qué hace cada elemento y cómo se utiliza. No necesita acceder al código fuente.

Debido a actualizaciones, correciones, etc se necesita automatizar el proceso.

Herramientas que permiten generar documentación de forma automática a partir del código fuente:

- **Javadoc** herramienta estándar para Java.

- **phpDocumentor** herramienta para PHP.

Los entornos de programación modernos como **NetBeans** o **Eclipse** aprovechan los comentarios de nuestro código fuente para mostrar información muy útil.

Estas herramientas, esperan el mismo tipo de comentarios, basado en el **estándar** establecido por JavaDoc, de modo que haremos el trabajo una sola vez y podremos aprovecharnos del mismo en varios entornos y con varias herramientas.

## 2.- PhpDocumentor.

Herramienta más utilizada para php.

La documentación de un proyecto de software es tan importante como su código.

Una buena documentación nos facilita, en gran medida, el mantenimiento futuro de la aplicación.

Y de gran utilidad si trabajamos en equipo.

phpDocumentor, genera automáticamente documentación de nuestro código, mediante comentarios y unas etiquetas especiales podemos definir de forma sencilla qué hace cada clase, cada método y cada función de nuestro código.

phpDocumentor (software libre) permite generar la documentación de varias formas y en varios formatos.

* Desde línea de comandos (CLI).
* Desde interfaz web.
* Desde código, scripts.
* Integrado con los IDEs de desarrollo en PHP (NetBeans, VSC, Eclipse).

Parámetros específicos:

1. El directorio en el que se encuentra nuestro código.
2. Opcionalmente los paquetes (@pakage) que deseamos documentar, lista de ficheros incluidos y/o excluidos y otras opciones interesantes para personalizar la documentación.
3. El directorio en el que se generará la documentación.
4. Si la documentación va a ser pública (sólo interfaz) o interna (en este caso aparecerán los bloques private y los comentarios @internal).
5. El formato de salida de la documentación.

Formato de salida:

* **HTML** a través de un buen número de plantillas predefinidas.
* **PDF**
* **XML** (DocBook), a partir de este podemos transformar (XSLT) a cualquier otro utilizando nuestras propias reglas y hojas de estilo.

Alternativa **Doxygen** la diferencia principal es que es un programa, mientras que phpDocumentor es una colección de código en PHP es por eso que se necesita tener PHP instalado.

## 2.1.- Instalación de phpDocumentor

* Instalación de Ubuntu 20.04 o superior
* Instalación de Xampp
* Hacer una prueba: <http://localhost>
* CLI: php –r “phpinfo();”
* En /opt/lampp/htdocs : crear phpinfo.php
* Descargar el paquete mediante: wget <https://phpdoc.org/phpDocumentor.phar>
* Luego:
  + Chmod +x phpDocumentor.phar
  + Sudo mv phpDocumentor.phar /usr/local/bin/phpdoc
* Manera global: phpdoc –d . –t docs/api
* Este comando genera una estructura de documentación vacía en la carpeta docs/api.

## 2.2.-Funcionamiento de phpDocumentor

La documentación se distribuye en bloques “DocBlock”. Estos bloques siempre se colocan justo antes del elemento al que documentan y su formato es:

|  |
| --- |
| /\*\*  \* texto  \* mas texto  \*/  function suma(){  } |

Los elementos que pueden ser documentados son:

* Function
* Costant
* Class
* Interface
* Trait
* Class constant
* Property
* Method

Se puede incluir documentación global a nivel de fichero y clase mediante **@package.**

Dentro de cada **DockBlock** se pueden incluir marcas o etiquetas:

* @author
* @copyright
* @deprecated
* @example: permite especificar la ruta hasta un fichero con código PHP.
* @ignore: evita que phpDocumentor documente un determinado elemento.
* @internal: documentación interna, NO es pública.
* @link: enlace <http://> a un determinado recurso.
* @see:crea enlaces internos.
* @since: indica que el elemento está disponible desde una determinada versión del paquete o distribución.
* @version

Marcas que solamente se pueden utilizar en bloques de determinados elementos:

* @global: para especificar el uso de variables globales dentro de una función.
* @param: para documentar parámetros que recibe una función.
* @return: valor devuelo por una función.
* @throws: indica si el método o la función puede lanzar algún tipo de excepción.

Crear proyecto.

Generar la documentación de manera automática: phpdoc –d . –t doc

La documentación se generará y almacenará en la carpeta doc. <http://localhost/banco/doc>

## 2.3.-Configuración de phpDocumentor