Introducción a la Documentación en PHP

Es usted de los que codifica y no realiza la documentación? , Le ha tocado modificar el código de otra persona el cual no entiende, porque no entiende los tipos de parámetros que ingresan, que tipo de retorna?, Ha modificado su código el cual no se acuerda de que hace porque hace rato no miraba el código?, Entonces para esto lo invitamos a que documente su código fuente. Por eso en este artículo, haremos una pequeña introducción para documentar su código fuente para desarrolladores en PHP.

Docblocks

Un Docblock no es más que un bloque de documentación estilo C o C-stlye para documentar un bloque de código. Comienza con / ** y tiene un asterisco al principio de cada línea. He aquí un ejemplo:

?

```
/**
 * Calcula la suma del cuadrado de un arreglo
 *
 * Ciclo que recorre el arreglo, obtiene el valor lo eleva
al cuadrado y se lo
 * suma y retorna el total
 *
 * @param array $arr
 * @return int
 * @throws Exception Si el elemento no es un entero
 */
function sumaDeCuadrados($arr) {
$total = 0;
foreach ($arr as $val) {
if (!is_int($val)) {
throw new Exception("El elemento no es un entero !");
}
$total = $val * $val;
}
return $total;
}
```

El DocBlocks consiste en tres partes todas tres opcionales, la descripción corta, la descripción larga y los tags. La descripción corta, la cual resumen la funcionalidad, la descripción larga la cual explica de manera específica el funcionamiento el cual puede ser tan largo como desee. Los tags de los DocBlocks contiene una serie de etiquetas especiales indicados por el símbolo @. Los tags se utilizan para especificar información adicional, tales como los parámetrosesperados y su tipo.

No todo el código se puede documentar con DocBlocks, acá están la lista de elemento que deberían ser documentados

- Archivos
- Clases
- Funciones y métodos

- Propiedades de las clases
- Variables Globales
- include()/require()
- define()

Archivos / Files

En los Docblocks se documentan el contenido de un archivo, la mayoría de elementos están documentados mediante la colocación del Docblocks antes del elemento, pero los archivos son una excepción (ya que no puede escribir nada antes de un archivo). A nivel de archivo docblocks se colocan en la primera línea del archivo.

Para este tipo de docblocks es recommendado utilizar las siguientes tags: **@package**, **@subpackage**, **@license**, y **@author**. De los tags hablaremos mas adelante.

Aca un ejemplo de documentación de un file:

?

```
/**
 * Este Archivo contiene funciones utilizadas dentro del programa
 *
 * @package    MiProyecto
 * @subpackage Comun
 * @license http://opensource.org/licenses/gpl-license.php  GNU Public License
 * @author     Pedro Picapiedra <pedropicapiedra@yabadabado.com>
 */
```

Clases

Un DocBlock de clase es colocado siempre antes de la clase, en esta va la descripción de la clase, Generalmente utiliza los

tags **@package**, **@subpackage**, and **@author**. Por Ejemplo :

?

```
1  /**
2   * Un  ejemplo de clase
3   *
4   * Colocaremos la clase vacia para el ejemplo
5   *
6   * @package    MiProyecto
7   *  @subpackage Comun
8   * @author     Pedro Picapiedra <
    pedropicapiedra@yabadabado.com>
9   */
10  class Example {
11  }
```

Métodos o funciones

Los métodos y las funciones se documentan igual, para los que no saben que es un método, el método es una función, pero esta, se encuentra dentro de una clase. Funciones y métodos generalmente contiene los tags **@param** y **@return**, estos tags nos indicaran el tipo de datos de entrada y el tipo de datos de salida. Ejemplo :

?

```
1   /**
2    * la división de dos numeros
3    *
4   * @param int $a primero numero ingresado
5    * @param int $b Segundo numero entrado
6    * @return int  Retorna
7    */
8   function division($a,  $b){
9   if ($b  > 0) {
10  return $a/$b;
11  }
12  return 0;
13  }
```

Defines/ Includes / Require

Los DocBlocks para estos elementos son en general los mismo simplemente se necesita una descripción, de lo que estas funciones puede realizan. Ejemplo

?

```
1                       /**
2 * Me define la variable global para el ejemplo
3                       */
4          define("VARIABLE_GLOBAL, 0);
```

Tags

La utilización de tags, varía según el tipo de DocBlocks, que se utiliza, por eso hablaremos de los tags más comunes y de los cuales utilizamos en nuestros ejemplos.

@package:se utilizan para agrupar elementos relacionados con el código en la documentación generada. Puede especificar los paquetes de los archivos y las clases y el código documentado que contienen van a heredar. Ejemplo

?

```
1                   /**
2 * Un Bloque de documentación para algo
3          * @package Miproyecto
4                   */
```

@subpackage: Si usted tiene múltiples niveles de ***@package*** entonces ***@subpackage*** puede definir un paquete de sub paquete ejemplo:

?

```
1                   /**
2 * Un Bloque de documentación para algo
3          * @package Miproyecto
4        * @subpackage subpaquete
5                   */
```

@license: Este es usada para cualquier include, page, class, function, define, method, variable y que indique la url de su licencia de uso. Ejemplo

?

```
1
2
3
4
/**
 * @license http://opensource.org/licenses/gpl-license.php
                    GNU Public License
                                */
                class opensource_class {...}
```

@author Este tag es usada para indicar el autor de cualquier elemento include, page, class, function, define, method, variable, ejemplo:

?

```
1
2
3
4
5
/**
  * Ejemplo de author en un docblock
* @author Pablo Marmol <pmpdka@php.net>
                    */
      Function datafunction(){….}
```

@param : Este tag es para indicar los parámetros de entrada en una función o método, el cual indicara el tipo de parámetro (int, string, bool, etc), seguido por el nombre de la variable y por ultimo una pequeña descripción.

En algunos casos el parametro puede ser de diferentes tipos, por lo cual separamos los diferentes tipos por una línea vertical o pipe |.

@return:se utiliza para documentar el valor de retorno de funciones o métodos. Al igual que los parámetros los tipos son (int, string, bool, etc), y puede ser multiples y se delimitan igual. **Ejemplo para ambos casos:**

?

```
/**
 * Finds and returns user by ID or username
 *
* @param int|string $usuario Identificador de usuario o El nombre de usuario
 * @return string Numero de document del usuario
 */
function obtieneNumero($user)
{
// ...
if ($isFound) {
return new User(...);
}
return null;
}
```

Para este artículo solo hemos colocado los tags más comunes, claro está que existe otro tipo de tags utilizados y de gran uso para mejor la documentación del codigo fuente.

Para finzalizar podemos agregar que la documentación, del codigo fuente, le ayudara a usted o a su equipo de desarrollo mejor el codigo fuente, tenerlo mas organizado y que cualquier programador pueda ingresar y entender lo que usted como autor quizo plasmar en el codigo, como recomendación final, para plasmar la documentación del codigo fuente en formatos mas agradables, los invitamos a revizar los programas phpDocumentor y DocBlox, para que les genere un documento organizado y estándar para que sirva como manual del programador.

fuente:di.sld.cu