Published on *Marco de Desarrollo de la Junta de Andalucía* (http://madeja.i-administracion.junta-andalucia.es/servicios/madeja)

Convenio de codificación específico para Drupal

- **Área:** Especificaciones de Codificación y Construcción
- **Tipo de pauta:** Directriz
- **Carácter de la pauta:** Obligatoria
- **Tecnologías:** PHP

Código: LIBP-0124

Se recogen una serie de convenios de codificación específicos de Drupal, a tener en cuenta además de los generales

La comunidad Drupal ha considerado necesario la introducción de una serie de estándares que faciliten la legibilidad y estructuración del desarrollo de código para Drupal. A continuación se van a ofrecer una serie de indicaciones para la codificación php en Drupal.

Pautas

Título	Carácter
Etiquetas de apertura y cierre en PHP	Obligatoria
Etiqueta de cierre al final del código de los archivos	Obligatoria
Sangría de lineas	Obligatoria
Declaración de funciones	Obligatoria
Parámetros por defecto	Obligatoria
Operadores	Obligatoria
Casting	Obligatoria
Longitud de líneas superior a 80 caracteres	No Recomendada
Manejo del array	Obligatoria
Llamadas al método constructor de la clase	Obligatoria
Concatenación de cadenas	Obligatoria
Punto y coma	Obligatoria
Nomenclatura de clases y métodos	Obligatoria
Nomenclatura de funciones	Obligatoria
Nomenclatura de los ficheros	Obligatoria
Nomenclatura de los módulos	Obligatoria
Doxygen en Drupal	Recomendada

Etiquetas de apertura y cierre en PHP

Utilizar <?php y ?> para delimitar el código PHP

En Drupal se utilizan los siguientes principios para el manejo de las etiquetas de apertura y cierre de código PHP:

- Utilice **<?php ?>** para delimitar el código PHP
- La etiqueta de apertura simplificada, **<? ?>** nunca debe usarse

Volver al índice

Etiqueta de cierre al final del código de los archivos

No utilizar la etiqueta de cierre ?> al final del código de los archivos

Nota: a partir de Drupal 4.7, el **"?>"** al final del código de los archivos, es omitido deliberadamente.

Las razones para esto se pueden resumir como:

- Elimina la posibilidad de espacios en blanco indeseados al final de los archivos, lo cual puede causar errores "header already sent" (cabecera ya enviada), problemas de validación XHTML/XML y otros problemas.
- El delimitador de cierre al final de un archivo es opcional
- PHP.net elimina por sí mismo el delimitador de cierre al final de sus archivos (ejemplo:prepend.inc), por lo que puede ser visto como una "buena práctica".

Volver al índice

Sangría de lineas

Usar una sangría de 2 espacios en blanco, sin tabulaciones

El código debe tener **2 espacios en blanco** para la sangría, que no sean tabulaciones.

Volver al índice

Declaración de funciones

Los nombres de las funciones deben incluir el nombre de grupo

Las funciones deben tener el nombre del grupo o módulo como prefijo, para evitar conflictos de nombres entre funciones de distintos módulos.

Volver al índice

Parámetros por defecto

Listar los parámetros por defecto de las funciones

Al escribir una función que utilice valores por defecto para algunos parámetros, deben listarse estos parámetros.

Volver al índice

Operadores

Colocar un espacio en blanco antes y después de cada operador binario

Todos los operadores binarios (operadores que están entre dos valores), como +, -, +=, !=, ==, >, etc. deben tener un espacio antes y después del operador, para facilitar la lectura.

Volver al índice

Casting

Colocar un espacio entre el (tipo) y la $variable en una transformación

Colocar un espacio entre el (tipo) y la $variable en una transformación. Por ejemplo, (int) $mynumber

Volver al índice

Longitud de líneas superior a 80 caracteres

Salvo las excepciones indicadas, las líneas de código no deben tener más de 80 caracteres

- Las líneas que contengan nombres largos de funciones, definiciones de función/clase, declaraciones de variable, etc pueden superar los 80 caracteres.
- Las condiciones de las estructuras de control pueden exceder los 80 caracteres, si es que son simples de leer y entender. Las condiciones no deben ser encapsuladas en líneas múltiples. En cambio, es una práctica recomendada dividir y preparar las condiciones por separado, lo cual también permite documentar las razones subyacentes de las condiciones

Volver al índice

Manejo del array

Los arrays deberán estar formateados y cada elemento del array debe moverse a su propia línea si el bloque del array tiene más de 80 caracteres

Los arrays deberán estár formateados con espacios separados para cada elemento y cada operador de asignación. Si el bloque de un array tiene más de 80 caracteres, es una buena práctica para facilitar la legibilidad y mantenibilidad mover cada

elemento a su propia línea.

Volver al índice

Llamadas al método constructor de la clase

Incluir siempre los paréntesis cuando se llama a un método constructor.

Cuando se llame a un método constructor sin argumentos, deben incluirse siempre en la llamada los paréntesis del método.

Volver al índice

Concatenación de cadenas

Utilizar un espacio entre el punto y las partes concatenadas

Emplear siempre un espacio entre el punto y las partes concatenadas de una cadena para mejorar la legibilidad. Al concatenar variables simples se pueden usar comillas dobles y agregar la variable dentro. En otro caso, es necesario usar comillas simples. Al usar el operador de “concatenación-asignación” (‘.=’), también es necesario incluir un espacio antes y después del operador.

Volver al índice

Punto y coma

Utilizar siempre el punto y coma (;) al final de cada línea, incluso al final de los bloques de código

El lenguaje PHP requiere puntos y comas al final de la mayoría de las líneas, pero permite ser omitidos al final de bloques de código. Los Estándares de programación de Drupal los requieren, incluso al final de bloques de código.

Volver al índice

Nomenclatura de clases y métodos

Utilizar la nomenclatura "CamelCase" para nombrar las clases

Las clases deben ser nombradas usando la nomenclatura "**CamelCase**." Los métodos y propiedades de clases deben usar "**lowerCamelCase**"

Volver al índice

Nomenclatura de funciones

Nombrar las funciones en minúscula, basándose en el nombre del módulo

Los nombres de las funciones estarán en minúsculas y basados en el nombre del módulo del que forman parte. Los guiones bajos se utilizarán para separar las palabras descriptivas del nombre. Después del nombre del módulo, la función debe nombrarse con el verbo y el objeto a los que hace referencia.

Volver al índice

Nomenclatura de los ficheros

Escribir los nombres de los ficheros en minúscula

Los nombres de los ficheros deben estar en minúsculas. La excepción es para los ficheros de documentación, que deben escribirse en mayúsculas y con la terminación ".txt"

Volver al índice

Nomenclatura de los módulos

No usar guiones bajos para el nombre de los módulos

El nombre de un módulo no debe contener guiones bajos para evitar conflictos de espacios de nombres.

Volver al índice

Doxygen en Drupal

Usar Doxygen para generar la documentación de Drupal

Se recomienda usar la herramienta Doxygen para generar la documentación de Drupal ya que permite elegir entre una gran

variedad de formatos para generar la documentación.

Volver al índice

Pautas

Área: Desarrollo » Especificaciones de Codificación y Construcción

Código	Título	Tipo	Carácter
LIBP-0008	Convenio de codificación general	Directriz	Obligatoria

Recursos

Área: Desarrollo » Especificaciones de Codificación y Construcción

Código	Título	Tipo	Carácter
RECU-0620	Implementación de convenios de codificación en Drupal	Referencia	Obligatorio
RECU-0110	Doxygen	Referencia	Permitido