***Estándar de codificación básica PSR-1***

***1. Visión general***

* Los archivos DEBEN utilizar solamente las etiquetas <?php y <?=.
* Los archivos DEBEN emplear solamente la codificación UTF-8 sin BOM para el código PHP.
* Los archivos DEBERÍAN declarar cualquier estructura (clases, funciones, constantes, etc,...) o realizar partes de la lógica de negocio (por ejemplo, generar una salida, cambio de configuración ini, etc,...) pero NO DEBERÍAN hacer las dos cosas.
* Los espacios de nombres y las clases DEBEN cumplir el estándar PSR-0.
* Los nombres de las clases DEBEN declararse en notación StudlyCaps. [^1]
* Las constantes de las clases DEBEN declararse en mayúsculas con guiones bajos como separadores CONSTANTE\_DE\_CLASE.
* Los nombres de los métodos DEBEN declararse en notación camelCase. [^2]

**Notas:**

**Tapas Studly** es una forma de notación de texto en el que la capitalización de las letras varía según algún patrón, o arbitrariamente, por lo general también omitiendo espacios entre palabras y menudo omitiendo algunas letras, por ejemplo, StUdLyCaPs o STuDLyCaPS. Estos patrones son identificados por muchos usuarios, ambiguamente, como el caso de camellos. La alternativa típica es simplemente reemplazar los espacios con guiones bajos (como en el caso de la serpiente).

**CamelCase** es un estilo de escritura que se aplica a frases o palabras compuestas. El nombre se debe a que las mayúsculas a lo largo de una palabra en CamelCase se asemejan a las jorobas de un camello. El nombre CamelCase se podría traducir como Mayúsculas/Minúsculas Camello. El término case se traduce como "caja tipográfica", que a su vez implica si una letra es mayúscula o minúscula y tiene su origen en la disposición de los tipos móviles en casilleros o cajas.

Existen dos tipos de CamelCase:

**UpperCamelCase**, cuando la primera letra de cada una de las palabras es mayúscula. Ejemplo: EjemploDeUpperCamelCase.

**lowerCamelCase**, igual que la anterior con la excepción de que la primera letra es minúscula. Ejemplo: ejemploDeLowerCamelCase.

***2. Archivos***

***2.1. Etiquetas PHP***

El código PHP DEBE utilizar las etiquetas largas <?php ?> o las etiquetas cortas para imprimir salida de información <?= ?>; NO DEBE emplear otras variantes.

***2.2. Codificación de caracteres***

El código PHP DEBE utilizar codificación UTF-8 sin BOM.

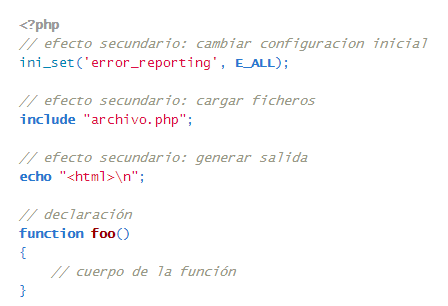
***2.3. Efectos secundarios***

Un archivo DEBERÍA declarar estructuras (clases, funciones, constantes, etc,...) y no causar efectos secundarios, o DEBERÍA ejecutar partes de la lógica de negocio, pero NO DEBERÍA hacer las dos cosas.

La frase "efectos secundarios" significa: que la ejecución de la lógica de negocio no está directamente relacionado con declarar clases, funciones, constantes, etc, simplemente la de incluir el archivo.

"Efectos secundarios" incluyen, pero no se limitan a: generar salidas, uso explícito de requiere o include, conexiones a servicios externos, modificación de configuraciones iniciales, enviar errores o excepciones, modificar variables globales o estáticas, leer o escribir un archivo, etc.

El siguiente ejemplo muestra un archivo que incluye las dos: declaraciones y efectos secundarios; Un ejemplo de lo que debe evitar:



El siguiente ejemplo es el de un archivo que contiene declaraciones sin efectos secundarios; Un ejemplo que puede seguir:

