

Recomendaciones de Mejores Prácticas de Codificación para lenguaje PHP

**Bogotá D.C., 10 de Septiembre de 2018**

Información del Documento

|  |  |
| --- | --- |
| **Nombre del Proyecto:** | Documentación de la Arquitectura Empresarial del Instituto Colombiano Agropecuario –ICA– |
| **Preparado por:** | Danny Ríos Tolosa | **Versión del Documento No:** | 0.4 |
| **Título:** | **Recomendaciones de Mejores Prácticas de Codificación para Lenguaje: PHP** | **Fecha de Versión del Documento:** | 2018-09-10 |
| **Revisado por:** |  | **Fecha de Revisión** | 2018-09-20 |

Registro de Cambios y Actualizaciones del Documento

| **Versión N°.** | **Fecha de Versión** | **Responsable del cambio** | **Descripción del Cambio** |
| --- | --- | --- | --- |
| 0.1 | 2018-03-11 | Danny Ríos Tolosa  | Versión inicial del documento |
| 0.2 | 2018-03-12 | Randolfo Gerardo Patiño  | Ajustes generales, para solicitud de aprobación. |
| 0.3 | 2018-05-29 | Randolfo Gerardo Patiño  | Ajustes al documento, por cambio de logo-símbolos institucionales |
| 0.4 | 2018-09-10 | Randolfo Gerardo Patiño  | Actualización documental. |
|  | *[AAAA/MM/DD]* |  |  |
|  | *[AAAA/MM/DD]* |  |  |

TABLA DE CONTENIDO

Pág.

[1. Introducción 5](#_Toc508662065)

[2. Convenciones del documento 5](#_Toc508662066)

[3. Resumen de estándares y codificación 6](#_Toc508662067)

[3.1. Palabras reservadas 6](#_Toc508662068)

[3.2. Estándar de codificación 7](#_Toc508662069)

[3.2.1. Estándar de codificación básica 7](#_Toc508662070)

[3.2.1.1. Archivos PHP 7](#_Toc508662071)

[3.2.1.2. Constantes en clases, Propiedades y Métodos 7](#_Toc508662072)

[3.3. Estilo de codificación 7](#_Toc508662073)

[3.3.1. General 7](#_Toc508662074)

[3.3.2. Manejo de cadenas de caracteres 8](#_Toc508662075)

[3.3.3. Manejo de matrices multidimensionales (Array) 9](#_Toc508662076)

[3.3.4. Declaración de *“namespaces”* y clausula *“use”* 9](#_Toc508662077)

[3.3.5. Clases, Propiedades y Métodos 10](#_Toc508662078)

[3.3.6. Estructuras De Control 14](#_Toc508662079)

[REFERENCIA BIBLIOGRÁFICA 17](#_Toc508662080)

LISTA DE TABLAS

Pág.

[Tabla 1. URL de las fuentes de información de referencia 5](#_Toc508662093)

[Tabla 2. Convenciones de Palabras Clave 6](#_Toc508662094)

[Tabla 3. Palabras Reservadas 7](#_Toc508662095)

LISTA DE ILUSTRACIONES

[Ilustración 1. Formatos Cadena de Texto con Sustitución de Variable 8](#_Toc508662163)

[Ilustración 2. Concatenación de Cadenas de Caracteres 9](#_Toc508662164)

[Ilustración 3. Declaración array Asociativo 9](#_Toc508662165)

[Ilustración 4. Declaración Alternativa Array Asociativo 9](#_Toc508662166)

[Ilustración 5. Declaración de *“namespaces”* y clausula *“use”* 10](#_Toc508662167)

[Ilustración 6. Disposición Llaves Clase 10](#_Toc508662168)

[Ilustración 7. Lista De Interfaces 11](#_Toc508662169)

[Ilustración 8. Declaracion de una propiedad 11](#_Toc508662170)

[Ilustración 9. Declaración Estándar De Un Método 12](#_Toc508662171)

[Ilustración 10. Declaración Alterna De Un Método 13](#_Toc508662172)

[Ilustración 11. Llamado Estandar De Función o Método 13](#_Toc508662173)

[Ilustración 12. Llamado Alterno De Función o Método 14](#_Toc508662174)

[Ilustración 13. Declaración Estructura de Control IF 14](#_Toc508662175)

[Ilustración 14. Declaración Estructura de Control SWITCH 15](#_Toc508662176)

[Ilustración 15. Declaración Estructura de Control WHILE 15](#_Toc508662177)

[Ilustración 16. Declaración Estructura de Control DO WHILE 15](#_Toc508662178)

[Ilustración 17. Declaración Estructura de Control FOR 16](#_Toc508662179)

[Ilustración 18. Declaración Estructura de Control FOREACH 16](#_Toc508662180)

[Ilustración 19. Declaración Estructura de Control TRY, CATCH 16](#_Toc508662181)

**RECOMENDACIONES DE MEJORES PRÁCTICAS DE CODIFICACIÓN PARA LENGUAJE PHP**

# Introducción

El presente documento describe las normas para el desarrollo de aplicaciones web usando lenguaje de programación de plataforma libre PHP, y su objetivo es tomar un estándar que haya sido validado por la industria del desarrollo de software, adoptarlo y ajustarlo a las necesidades del Instituto Colombiano Agropecuario; con el fin de crear sistemas de información de calidad guiados por los estándares aceptados por una de las comunidades más grandes del desarrollo web.

De forma específica se cubren temas como autocarga de dependencias, interfaces y estilos de codificación entre otros temas que serán tratados a continuación.

Las fuentes de información utilizadas para el desarrollo de este documento son las siguientes:

|  |  |
| --- | --- |
| **Tema Fuente de la Información** | **URL** |
| The PHP Framework Interoperability Group | https://www.php-fig.org |
| PHP Oficial Page | http://php.net/ |

Tabla 1. URL de las fuentes de información de referencia

*Fuente: PHP Standards Recommendations.*

# Convenciones del documento

El presente documento manejara las palabras clave "DEBE", "NO DEBE", "REQUERIDO", "DEBERÁ", "NO DEBERÁ", "DEBERÍA", "NO DEBERÍA", "RECOMENDADO", "PUEDE", y "OPCIONAL" para ser interpretado como se describe en RFC 2119[[1]](#footnote-2).

| ***Palabras Clave*** |
| --- |
| **Debe, Requerido, Deberá** | Cumplimiento absoluto de la especificación. |
| **No Debe, No Deberá** | Prohibición absoluta de la especificación. |
| **Debería, Recomendado** | Enfatiza que la regla seria bueno cumplirse siempre y cuando sea posible y sea adecuado |
| **No Debería, No Recomendado** | Enfatiza que la regla no sería bueno cumplirse siempre y cuando sea posible y sea adecuado |
| **Puede, Opcional** | Verdaderamente opcional |
| **CamelCase** | Forma de escribir una palabra donde su primera letra está en minúsculas, y la primera letra de las subsiguientes palabras en mayúsculas. |
| **StudlyCaps** | Forma de escribir una palabra donde la primera letra está en mayúscula, y la primera letra de las subsiguientes palabras igualmente en mayúsculas. |

Tabla 2. Convenciones de Palabras Clave

*Fuente: PHP Standards Recommendations.*

# Resumen de estándares y codificación

## Palabras reservadas

Las palabras reservadas en un lenguaje son utilizadas por el compilador con fines específicos, Las cuales no se deben utilizar al momento de definir los nombres de los elementos del programa tal como variables y procedimientos.

Las siguientes palabras son reservadas, no deben ser utilizadas para el nombramiento de variables, procedimientos y otros elementos

|  |
| --- |
| **Palabras Reservadas** |
| abstract | and | array | as | break |
| callable | case | catch | class | clone |
| const | continue | declare | default | die |
| do | echo | else | elseif | empty |
| enddeclare | endfor | endforeach | endif | endswitch |
| endwhile | eval | exit | extends | final |
| finally | for | foreach | function | global |
| goto | if | implements | include | include\_once |
| instanceof | insteadof | interface | isset | list |
| namespace | new | or | print | private |
| protected | public | require | require\_once | return |
| static | switch | throw | trait | try |
| unset | use | var | while | xor |
| yield |  |  |  |  |

Tabla . Palabras Reservadas[[2]](#footnote-3)

## Estándar de codificación

### Estándar de codificación básica

### Archivos PHP

* El código PHP **DEBE** usar las etiquetas largas <?php ?> o las etiquetas cortas de impresión <?= ?>; **NO DEBE** usar ninguna otra variación de etiquetas.
* El código PHP **DEBE** usar solo UTF-8 sin BOM.
* Un archivo **DEBERÍA** declarar nuevos símbolos (clases, funciones, constantes, etc.) y no usarse para otros efectos, o **DEBERÍA** ejecutar la lógica, pero **NO DEBERÍA** hacer ambas cosas.

### Constantes en clases, Propiedades y Métodos

* Las constantes de clase **DEBEN** declararse en mayúsculas con separadores de subrayado (\_).
* Para las propiedades se evita cualquier recomendación sobre el uso de StudlyCaps, camelCase o under\_score. Cualquiera que sea la convención que se defina en el proyecto, **DEBERÁ** ser aplicarse de manera coherente dentro del alcance razonable, ya que ese alcance puede llegar a ser a nivel del proveedor, paquete, clase o método.
* Los nombres de los métodos **DEBEN** declararse en camelCase.

## Estilo de codificación

### General

* El código **DEBE** seguir todas las reglas del apartado anterior
* Todos los archivos PHP **DEBEN** usar el final de línea Unix.
* Todos los archivos PHP **DEBEN** terminar con una línea en blanco.
* La etiqueta de cierre **DEBE** omitirse en los archivos que solo contengan PHP.
* **NO DEBE** haber un límite estricto en la longitud de la línea.
* Se **PUEDE** agregar líneas en blanco para mejorar la legibilidad e indicar bloques de código relacionados.
* **NO DEBE** haber más de una declaración por línea.
* El código **DEBE** usar una indentación de 4 espacios, y **NO DEBE** usar tabulaciones (TAB) para indentar.
* Las palabras clave de PHP **DEBEN** estar en minúsculas.
* Las constantes de PHP *true*, *false* y *null* **DEBEN** estar en minúsculas.

### Manejo de cadenas de caracteres

* Cuando una cadena es literal (no contiene sustituciones de variables), el apóstrofe o "comilla simple" **DEBE** ser utilizado para demarcar la cadena
* Cuando una cadena literal en sí misma contiene apóstrofes, **PUEDE** demarcar la cadena con comillas o "comillas dobles".
* La sustitución de variables **DEBERÍA** usar cualquiera de las siguientes formas.

[[3]](#footnote-4)

Ilustración . Formatos Cadena de Texto con Sustitución de Variable

* Las cadenas **DEBEN** concatenarse con el operador. (Punto), Siempre se debe agregar un espacio antes y después del operador para mejorar la legibilidad
* Cuando concatena cadenas con el operador (Punto), **DEBERÍA** dividir la declaración en múltiples líneas para mejorar la legibilidad. En estos casos, cada línea sucesiva **DEBERÍA** rellenarse con un espacio en blanco tal que el operador (Punto) quede alineado bajo el operador =



Ilustración . Concatenación de Cadenas de Caracteres

### Manejo de matrices multidimensionales (Array)

* Al declarar matrices indexadas, se **DEBE** agregar un espacio posterior después de cada coma para mejorar la legibilidad
* Se permite declarar matrices indexadas de múltiples líneas usando la construcción de la matriz. En este caso, cada línea sucesiva **DEBE** rellenarse con espacios tales que el comienzo de cada línea esté alineado con el elemento inicial de la matriz.
* Al declarar matrices asociativas con la construcción de la matriz, uno **DEBERÍA** dividir la declaración en múltiples líneas. En este caso, cada línea sucesiva **DEBE** rellenarse con un espacio en blanco de modo que tanto las claves como los valores estén alineados:

[[4]](#footnote-5)

Ilustración . Declaración array Asociativo

* Alternativamente, el elemento inicial de la matriz **PUEDE** comenzar en la siguiente línea. Si es así, **DEBE** rellenarse a un nivel de identación mayor que la línea que contiene la declaración de matriz, y todas las líneas sucesivas **DEBEN** tener la misma sangría; la llave de cierre **DEBE** estar en una línea sola al mismo nivel de identación que la línea que contiene la declaración de la matriz. Para la legibilidad, los diversos => operadores de asignación **DEBEN** ser indentados de forma que se alineen.

[[5]](#footnote-6)

Ilustración . Declaración Alternativa Array Asociativo

### Declaración de *“namespaces”* y clausula *“use”*

* Cuando está presente, **DEBE** haber una línea en blanco después de la declaración del *“namespace”*.
* Cuando están presentes, todas las declaraciones de *“use”* **DEBEN** ir después de la declaración del *“namespace”*.
* **DEBE** haber una palabra clave *“use”* por declaración.
* **DEBE** haber una línea en blanco después del bloque declaraciones de *“use*”.

[[6]](#footnote-7)

Ilustración . Declaración de *“namespaces”* y clausula *“use”*

### Clases, Propiedades y Métodos

* Las palabras clave *“extends”* e *“implements”* **DEBEN** declararse en la misma línea que el nombre de la clase.
* La llave de apertura para la clase **DEBE** ir en su propia línea; la llave de cierre de la clase debe ir en la línea siguiente después de que el cuerpo de la clase.

[[7]](#footnote-8)

Ilustración . Disposición Llaves Clase

* Las listas de *“implements”* **PUEDEN** dividirse en varias líneas, donde cada línea posterior se indenta una vez. Al hacerlo, el primer elemento de la lista **DEBE** estar en la siguiente línea, y **DEBE** haber solo una interfaz por línea.

[[8]](#footnote-9)

Ilustración . Lista De Interfaces

* La visibilidad **DEBE** declararse en todas las propiedades.
* **NO DEBE** haber más de una propiedad declarada por sentencia.
* Los nombres de propiedad **NO DEBEN** tener prefijo para indicar la visibilidad protegida o privada.

[[9]](#footnote-10)

Ilustración . Declaracion de una propiedad

* La visibilidad **DEBE** declararse en todos los métodos.
* Los nombres de los métodos **NO DEBEN** tener prefijo para indicar la visibilidad protegida o privada.
* Los nombres de los métodos **NO DEBEN** declararse con un espacio después del nombre del método. La llave de apertura **DEBE** ir en su propia línea, y la llave de cierre **DEBE** ir a la siguiente línea despues del cuerpo. **NO DEBE** haber un espacio después del paréntesis de apertura, y **NO DEBE** haber un espacio antes del paréntesis de cierre.
* En la lista de argumentos, **NO DEBE** haber un espacio antes de cada coma, y **DEBE** haber un espacio después de cada coma.
* Las listas de argumentos **PUEDEN** dividirse en varias líneas, donde cada línea subsiguiente se sangra una vez. Al hacerlo, el primer elemento de la lista **DEBE** estar en la siguiente línea, y **DEBE** haber solo un argumento por línea.
* Cuando la lista de argumentos se divide en varias líneas, el paréntesis de cierre y llave de apertura **DEBEN** ser colocados juntos en su propia línea con un espacio entre ellos.

[[10]](#footnote-11)

Ilustración . Declaración Estándar De Un Método

[[11]](#footnote-12)

Ilustración . Declaración Alterna De Un Método

* Al hacer llamado de un método o llamada de función, **NO DEBE** haber un espacio entre el método o nombre de función y el paréntesis de apertura, **NO DEBE** haber un espacio después del paréntesis de apertura, y **NO DEBE** haber un espacio antes del paréntesis de cierre. En la lista de argumentos, **NO DEBE** haber un espacio antes de cada coma, y **DEBE** haber un espacio después de cada coma.

[[12]](#footnote-13)

Ilustración . Llamado Estandar De Función o Método

* Las listas de argumentos **PUEDEN** dividirse en varias líneas, donde cada línea subsiguiente se indenta una vez. Al hacerlo, el primer elemento de la lista **DEBE** estar en la siguiente línea, y **DEBE** haber solo un argumento por línea.

[[13]](#footnote-14)

Ilustración . Llamado Alterno De Función o Método

### Estructuras De Control

* **DEBE** haber un espacio después de la palabra clave de la estructura de control.
* **NO DEBE** haber un espacio después del paréntesis de apertura.
* **NO DEBE** haber un espacio antes del paréntesis de cierre.
* **DEBE** haber un espacio entre el paréntesis de cierre y la llave de apertura.
* El cuerpo de la estructura **DEBE** estar indentado una vez
* La llave de cierre **DEBE** estar en la siguiente línea después del cuerpo
* El cuerpo de cada estructura **DEBE** estar rodeado por llaves. Esto estandariza el aspecto de las estructuras y reduce la probabilidad de introducir errores a medida que se agregan nuevas líneas al cuerpo.

[[14]](#footnote-15)

Ilustración . Declaración Estructura de Control IF

[[15]](#footnote-16)

Ilustración . Declaración Estructura de Control SWITCH

[[16]](#footnote-17)

Ilustración . Declaración Estructura de Control WHILE

[[17]](#footnote-18)

Ilustración . Declaración Estructura de Control DO WHILE

[[18]](#footnote-19)

Ilustración . Declaración Estructura de Control FOR

[[19]](#footnote-20)

Ilustración . Declaración Estructura de Control FOREACH

[[20]](#footnote-21)

Ilustración . Declaración Estructura de Control TRY, CATCH

# REFERENCIA BIBLIOGRÁFICA

* PHP Standards Recommendations. Por: Framework Interoperability Group. Tomado el 12 de Marzo de 2018. <https://www.php-fig.org>
* Zend Framework 2 Coding Standards. Por: Matthew Weier O'Phinney. Tomado el 12 de Marzo de 2018. <https://github.com/zendframework/zendframework/wiki/Coding-Standards>
1. http://www.ietf.org/rfc/rfc2119.txt [↑](#footnote-ref-2)
2. Tomado de http://php.net/manual/es/reserved.keywords.php [↑](#footnote-ref-3)
3. Tomado de https://github.com/zendframework/zendframework/wiki/Coding-Standards#strings [↑](#footnote-ref-4)
4. Tomado de https://github.com/zendframework/zendframework/wiki/Coding-Standards#associative-arrays [↑](#footnote-ref-5)
5. Tomado de https://github.com/zendframework/zendframework/wiki/Coding-Standards#associative-arrays [↑](#footnote-ref-6)
6. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-7)
7. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-8)
8. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-9)
9. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-10)
10. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-11)
11. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-12)
12. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-13)
13. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-14)
14. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-15)
15. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-16)
16. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-17)
17. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-18)
18. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-19)
19. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-20)
20. Tomado de https://www.php-fig.org/psr/psr-2/ [↑](#footnote-ref-21)