

Estándar de codificación

Proyecto Renova-Spa

23 de febrero de 2016

Sandy Pech

Utrm (Universidad tecnologica de la riviera maya)

Profesor a cargo CRISTIAN BERNAL RAMÍREZ

# **PSR-0**

# (Estándar de codificación a usar)

Actualmente en desuso. A partir del 10/21/2014 PSR-0 marco como obsoleto. En su lugar se recomienda hacer uso de PSR-4 como una alternativa.

A continuación se describen los requisitos obligatorios que se mantuvieron a partir del estándar PSR-0.

## **Nombre de archivos**

1. Los namespaces y las clases deben tener la siguiente estructura ()\*

\ <Nombre del proveedor> \ (<espacio de nombres> \) \* <nombre de clase>

1. Cada namespace debe tener un namespace superior ("Vendor name").

("Nombre Proveedor").

1. Cada namespace puede tener tantos sub-namespaces como se quiera.

amespacepackageClass\_Name=>path/to/project/lib/vendor/namespace/package/Class/Name.php

1. Los nombres de los namespaces o clases deben ser separados por un guion bajo (\_).

amespacepackageClass\_Name

1. Todos los archivos deben tener la extensión .php.

Name.php

1. Los nombres de los namespaces o clases deben ser ordenadas alfabéticamente.

## **Convenciones**

## Los archivos deben utilizar solamente

## Los archivos sólo deben utilizar una codificación UTF-8.

## Las funciones sólo deben retornar un sólo valor.

## Los nombres de espacio y las clases deben seguir las reglas del PSR-0.

## Los nombres de clases deben ser escritas utilizando la técnica StudlyCaps.

## Las constantes deben ser definidas en MAYÚSCULAS y utilizando guion bajo (\_) cómo separador.

## Métodos y funciones deben ser escritos utilizando la técnica camelCase.

## Debemos de validar que la función que vamos a crear no exista utilizando la función function\_exists().

## Las llaves deben de estar abajo sólamente en las clases y métodos.

## La identación debe ser con un tabulador establecido a 4 espacios.

## Las constantes true, false y null deben ser escritos en minúsculas.

## El número de caracteres por línea deben ser de 80 columnas aunque también esta aceptado que sean hasta 120.

## Ya no debes utilizar la palabra reservada var para declarar una propiedad, debes utilizar public, private, static o protected.

## Debe haber un espacio después de cada estructura de control (if, for, foreach, while, switch, try...catch, etc.).

# **PSR4**

# **Información general**

* Los archivos deben utilizar solamente ***<? Php***y ***<?*** = Etiquetas.
* Los archivos deben utilizar solamente ***UTF-8*** sin ***BOM*** para el código ***PHP***.
* Los archivos deben o bien declarar símbolos (clases, funciones, constantes, etc.) o causar efectos secundarios (por ejemplo, generar una salida, cambiar la configuración .ini, etc.), pero no debe hacer ambas cosas.
* Los espacios de nombres y clases deben seguir el estandar PSR "auto-carga": ***[***[***PSR-0***](https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-0.md)***,***[***PSR-4***](https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-4-autoloader.md)***].***
* Los nombres de clase deben ser declarados en StudlyCaps.
* constantes de clase se debe declarar en todo caso superior con separadores de subrayado.
* Los nombres de método se debe declarar en camelCase.

**Archivos**

### **Etiquetas**

Código PHP debe utilizar el tiempo **<? Php?>** Etiquetas o los cortos-eco **<=?>** Etiquetas; que no deberá utilizar las otras variaciones de la etiqueta.

### **Codificación de caracteres**

Código PHP sólo deberá utilizar UTF-8 sin BOM.

**Constantes**

Constantes de clase se debe declarar en todo caso superior con separadores de subrayado.**Por ejemplo:**

<?php

espacio de nombres de proveedores \ Modelo ;

clase Foo

{

const VERSION = ' 1.0 ' ;

const DATE\_APPROVED = ' 2012-06-01 ' ;

}

**Métodos**

Los nombres de método se debe declarar en camelCase () .