## Estándares de codificación

### Introducción

Este documento pretende definir punto claves sobre el formato del código fuente, el nombrado de archivos, ubicaciones y otros puntos clave para mantener un orden funcional, de congruencia y comprensión aceptable, y no ser una norma de carácter exhaustiva en implementación del sistema.

Se corresponde a un diseño estructural del tipo MVC, modelo – vista – controlador.

### Tipos de elementos

*Elemento fuente:* Archivo cuyo contenido es código fuente, el cual principalmente corresponderán a archivos de contenido PHP y/o recursos adicionales como archivos de datos (xml), de configuración (ini) y otros recursos relativos.

*Archivo de clase*: Es un *elemento fuente* cuyo contenido es estrictamente código PHP y solo tiene como contenido la definición de una clase.

*Archivo de modelo*: Es un a*rchivo de clase* el cual define una *clase de modelo*, es decir que interactúa con la base de datos.

*Archivo de controlador*: Es un *archivo de clase* el cual define una *clase de controlador*, la cual controlará las diferentes peticiones HTTP soportadas a través de la cual el sistema interactuará con sus usuarios.

*Archivo de ayudante*: Es un *archivo de clase* el cual define una *clase de ayudante*, compuesta únicamente por métodos de ayuda/soporte para el/los controladores.

*Archivo de bloque*: Es un *archivo de clase* el cual define una *clase de bloque*, la cual define el comportamiento para la recolección de datos de bloques de diseño utilizados para el despliegue gráfico en los *elementos de diseño*.

*Archivo de configuración de módulo*: Es un *elemento fuente,* es un tipo de archivo xml que contendrá las configuraciones del módulo, tales como asociación de paths de petición y clase controladora, temas implementados, definiciones de sub-módulos, declaración de *archivos de layout* introducidos por el módulo, etc.

*Elemento de diseño*: Son *elementos fuentes* que tengan la funcionalidad de generar pantallas (HTML), estos no interactuaran directamente con base de datos, ni realizan procesamiento de datos, simplemente se dedican a presentar datos de una manera gráficamente legible. También podrá utilizarse para referirse a plantillas de transformación xml (xslt), archivos de datos (xml), de configuración (ini), que sean utilizados para el despliegue visual.

*Archivo de template*: Son *elementos de diseño*, su contenido es mezcla de código PHP y HTML.

*Archivo de layout*: Es un e*lementos de diseños,* un tipo de archivo xml cuyo contenido define la disposición y configuración de los bloques a desplegar en las distintas pantallas del sistema.

*Recurso de diseño*: Corresponden a archivos cuyo contenido corresponden a imágenes, *plantillas de estilo*, *código javascript,* presentaciones flash y otros elementos eventualmente necesarios para el despliegue gráfico.

*Aplicación*: Es el nivel de modularización máximo de agrupación que define el sistema. Se implementarán inicialmente 2 tipos Backend y Frontend.

*Tema*: Es un nivel de modularizacion para *elementos de diseño* y *recursos de diseño* que presentará gráficamente con un estilo particular.

### Ubicaciones

La ubicación de archivos de código fuente estará ligada a el tipo de *elemento fuente* al que este pertenece. Tipos de elementos:

#### Elementos de diseño

La ubicación asignada (relativa al directorio raíz) es el directorio **“design”**. Este directorio a su vez estará divido en 2 niveles que organizaran dichos archivos.

El primer nivel corresponderá a la *aplicación* que al pertenece el *elemento de diseño. E*j.: design/frontend será el directorio donde se alojaran *elementos de diseño* para la *aplicación* Frontend.

El segundo nivel corresponderá al *tema* que al que se pertenece el *elemento de diseño.* Ej.: design/frontend/lime\_green/

#### Recursos de diseño

La ubicación asignada (relativa al directorio raíz) es el directorio **“skin”**. Este directorio a su vez estará divido en 2 niveles que organizaran dichos archivos.

El primer nivel corresponderá a la *aplicación* que al pertenece el *recurso de diseño. E*j.: skin/frontend será el directorio donde se alojaran *elementos de diseño* para la *aplicación* Frontend.

El segundo nivel corresponderá al *tema* que al que se pertenece el *recurso de diseño.* Ej.: skin/frontend/lime\_green/

#### Elemento fuente

La ubicación asignada para estos elementos será el directorio “**includes/classes**” y dentro del cual se subdividirá en cualquier cantidad de niveles, dependiendo del empaquetamiento necesario que se defina en el diseño del sistema. Ej.: includes/classes/Mascotas/ sería un directorio del paquete mascotas dentro del cual residirán los *elementos fuentes* para el paquete “Mascotas”.

#### Archivo de modelo

La ubicación asignada para estos elementos será el subdirectorio “**Model**” dentro del paquete al que pertenece la clase que define. Ej.: includes/classes/Mascotas/Model/Mascota.php

#### Archivo de controlador

La ubicación asignada para estos elementos será el subdirectorio “**Router**” dentro del paquete al que pertenece la clase que define. Ej.: includes/classes/Mascotas/Router/Search.php

En caso de un paquete solo contar con un controlador deberá nombrarse “Router” y estar en el directorio raíz del paquete. Ej.: includes/classes/Mascotas/Router.php

#### Archivo de ayudante

La ubicación asignada para estos elementos será el subdirectorio “**Helper**” dentro del paquete al que pertenece la clase que define. Ej.: includes/classes/Mascotas/Helper/Search.php

En caso de un paquete solo contar con un controlador deberá nombrarse “Helper” y estar en el directorio raíz del paquete. Ej.: includes/classes/Mascotas/Helper.php

#### Archivo de bloque

La ubicación asignada para estos elementos será el subdirectorio “**Block**” dentro del paquete al que pertenece la clase que define. Ej.: includes/classes/Mascotas/Block/SearchToolbar.php

#### Archivo de configuración de módulo

La ubicación asignada para estos elementos será el subdirectorio “**etc**” dentro del paquete al que pertenece la clase que define. Ej.: includes/classes/Mascotas/etc/config.xml

### Nomenclatura de Archivos

#### Tipos de nomenclaturas

**Camelcase**. Se refiere a la nomenclatura que solo contiene caracteres alfabéticos donde las mayúsculas delinean los comienzos de palabras. Ej.: TipoConsulta

**Underscored**: Se refiere a la nomenclatura compuesta únicamente por caracteres alfabéticos en minúsculas y guiones bajos que separan los comienzos de palabras. Ej.: tipo\_consulta.

**No restringida**: Se refiere a que no poseerá ninguna regla de nomenclatura.

\* se utilizará la simbología (p) para indicar que no es restrictivo, es decir que es una preferencia, en todos los otros casos su nomenclatura será restrictiva.

#### Tipos de elemento y nomenclatura

* Archivo de clase: Camelcase, extensión: php
* Archivo de configuración de módulo: config.xml
* Elemento de diseño: (p) Underscored
* Recurso de diseño: No restringida
* Elemento de template: extensión: phtml
* Paquete: Camelcase
* Tema: Underscored

### contenidos de elementos

#### Codificación PHP

\* **Atributos**: Son variables de instancia de clase.

##### Delimitación

Tanto para *archivos de clase* como para *archivos de template* el comienzo del código php dentro del contenido de los archivos será con el delimitador **<?php** y la finalización del mismo con el delimitador **¿>**.

No será aceptable su sintaxis alternativa corta donde el delimitador de comienzo es **<?**.

##### Nomenclatura

* **Clases**: Su nombre estará compuesto por una mezcla de nomenclatura camelcase y underscored, donde cada elemento que separan los guiones bajos deberá corresponder a la ruta de la clase y al nombre de su archivo. Ej.: includes/classes/Mascotas/Model/Mascota.php tendrá la definción de la clase Mascota\_Model\_Mascota.
* **Variables**: (p) underscored
* **Atributos**: underscored.
* **Métodos**: (p) Camelcase

#### Codificación Javascript

No se regulara ningún formato específico para este tipo de archivos.

#### Elementos XML

Se tabularan los elementos xml para una lectura mas sencilla del mismo. Ej:

<global>

<modules>

<Mascotas />

</modules>

<layouts>

<default>default.xml</default>

</layouts>

</global>

#### Plantillas de Estilo

Se utilizará para nombres de clases de estilo la nomenclatura underscored.

Preferentemente se podrán las reglas con el cuerpo abierto y tabulado. Ej.:

.destacado div{

height: 200px;

weight: 200px;

}