

## ESTÁNDARES DE PROGRAMACIÓN OFICINA VIRTUAL 2013

### 1. ESTÁNDARES PARA NOMBRES









- **UpperCamelCase**, cuando la primera letra de cada una de las palabras es mayúscula.
- **lowerCamelCase**, igual que la anterior con la excepción de que la primera letra es minúscula.

TIPO	NOMENCLATURA	OBSERVACIÓN
<b>Controlador (Controller)</b>	<b>[objeto]</b> Ejemplo archivo: persona, usuario, etc. Ejemplo clase: Persona, Usuario, etc.	Nombre del archivo: <b>lowerCamelCase</b> Nombre de la clase: <b>UpperCamelCase</b>
<b>Vista (View)</b>	<b>[accion _view] o [panel _view]</b>  Ejemplo archivo: ins_view, upd_view, etc.	Nombre del archivo: <b>lowerCamelCase</b> Sólo debe existir 4 vistas: - <b>ins_view</b> (insertar o nuevo) - <b>upd_view</b> (actualizar o editar) - <b>qry_view</b> (listado) - <b>panel_view</b> (contenido centralizado)
<b>Modelo (Model)</b>	<b>[objeto_model]</b> Ejemplo archivo: persona_model, usuario_model, etc. Ejemplo clase: Persona_model, Usuario_model, etc.	Nombre del archivo: <b>lowerCamelCase</b> Nombre de la clase: <b>UpperCamelCase</b>  <i>En el caso de necesitar métodos adicionales, por ejemplo: un combo para cargos, sería:</i> <b>cboCargosGet</b>
<b>Métodos (Methods)</b>	<b>[objetoMetodo]</b> Ejemplo método: personalns, usuarioUpd, personaGet (obtener un solo registro), personaQry (listado)	Nombre del método: <b>lowerCamelCase</b>  <i>Los métodos de controlador y el modelo tendrán el mismo estándar.</i>
<b>Carpetas (Fólders)</b>	<b>[Sistema o proyecto]</b> Ejem: sistram, sigma, etc	Solo minúsculas.
<b>Procedimientos Almacenados (Stored Procedures)</b>	<b>[USP_SIGLASISTEMA_TIPOOPERACIÓN_Objeto]</b> Ejm:  USP_SIT_S_Vehiculo USP_SIT_I_Vehiculo USP_SIT_U_Vehiculo	Todo en mayúscula a excepción del <b>Objeto</b> que sólo la primera letra es mayúscula  (Tipo de operación= U,I,S) • <b>U (Update)</b> • <b>I (Insert)</b> • <b>S (Select, cualquier lectura de la BD)</b>
<b>Controles XHTML</b>	<b>Ejemplos formulario insertar y actualizar</b> Nombre formulario : frm_ins_persona Caja texto para nombre : txt_ins_per_nombre Combo sexo de persona : cbo_ins_per_sexo Radio button de sexo : rdo_ins_per_sexo Check de intereses : chk_ins_per_intereses Botón registrar : btn_ins_per Dialogo de un form de edición : c_frm_upd_persona (div que contiene al form) Dialogo que contiene un combo : c_cbo_upd_per_sexo (div que contiene el combo)  <b>Ejemplos Formulario búsqueda</b> Caja de búsqueda : txt_fnd_per_dni Botón de búsqueda : btn_fnd_per_dni Combo en el buscar : cbo_fnd_per_sexo  <b>Anchor</b> Enlace ver datos persona: anc_fnd_per_datos	Se utilizará como ejemplo un CRUD para persona.  <b>[tipocontrol_accion_entidad_campodelform]</b>  <b>acn = anchor (enlace)</b> <b>fnd = find (buscar)</b> <b>c = container (contenedor)</b>  <i>En el caso de nombres compuestos, por ejemplo: fecha de nacimiento, sería:</i> <i>txt_ins_per_fecNac</i> (notación <b>lowerCamelCase</b> con 3 primeras letras por cada nombre).

JavaScript	<p><b>[abrevObjeto]</b>  <b>Ejem:</b> jsPersona, jsUsuario, etc.</p> <p>Para la realización de envíos POST o GET a un controlador u otro se utilizará:</p> <pre>\$.ajax({   type: "&lt;POST o GET&gt;",   url: "&lt;a donde enviar&gt;",   cache: false,   data: {     item:\$("#item").val(),     accion:&lt;accion&gt;"   },   success: function(data) {    },   error: function() {    } });</pre>	<p>Nombre del archivo: <b>lowerCamelCase</b></p> <p><i>En el caso de necesitar usar otro documento javascript (en el caso de un update), el document debería ser: jsPersona<b>Upd</b></i></p>
Tablas (SQL SERVER)	<p><b>Tabla :</b></p> <p><b>[objeto]</b></p> <p><b>Ejem :</b></p> <p><b>USUARIO, PERSONA, VEHICULO, etc</b></p> <p><b>Campos :</b></p> <p><b>[TipodeDato] (c,d,b,n) +</b>  <b>[TresPrimerasLetrasObjeto] +</b>  <b>[NombreCampo]</b></p> <p><b>Ejem :</b></p> <p><b>nUsulD,cPerNombres,bPerEliminado,dVehFechaRegistro</b></p>	<p><b>Tabla</b></p> <p>En Nombre de la tabla debe estar de preferencia en <b>singular</b> y en <b>mayúsculas</b></p> <p><b>Campos</b></p> <p><b>TipoDato:</b></p> <p><b>c:</b> carácter  <b>d:</b> fecha  <b>b:</b> boolean  <b>n:</b> numérico</p> <p>La Letra que identifica al tipo de dato es minúscula, las que identifican a la tabla es notación <b>UpperCamelCase</b> y las que identifican el nombre del campo también lleva notación <b>UpperCamelCase</b></p>

## 2. ESTÁNDARES PARA ÍCONOS

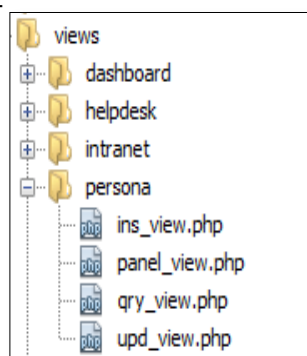
- Se debe utilizar los íconos de jQueryUI

DESCRIPCIÓN	ICONO NORMAL	ICONO HOVER
Nuevo		
Editar		
Eliminar		
Dar de Baja		



## 3. ESTÁNDARES PARA LOS NOMBRES DE ARCHIVOS

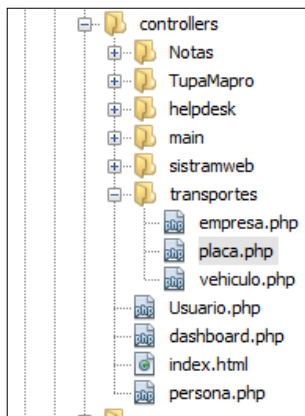
### a. VISTAS (VIEWS)



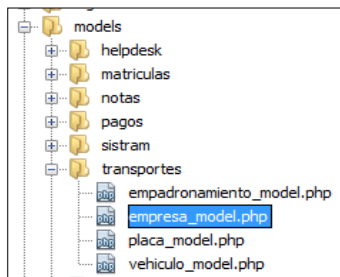
- Toda vista que pertenezca a un sistema en particular deberá ser incluida dentro de una carpeta.

- También será necesario crear carpetas para cada uno de los objetos y estas a su vez deberán tener 4 archivos básicos como se observa en la imagen de la izquierda.

## b. CONTROLADORES (CONTROLLERS)

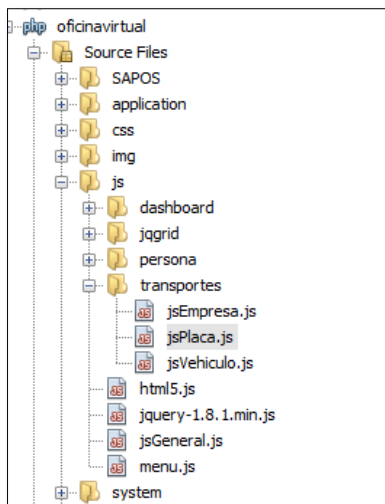


## c. MODELOS (MODELS)



En la carpeta **models** las únicas carpetas que deben existir son aquellas con el nombre del sistema.

## d. JAVASCRIPT (JS)



## Consideraciones importantes !

- Utilizar transacciones cuando se hace varios inserts a la vez (en CI : batch).
- Evitar, en lo posible, utilizar los eventos nativos de javascript, priorizando los eventos propios de jQuery.
- Para hacer referencia a las acciones (insertar, eliminar, actualizar, buscar, etc.) se utilizará una variable llamada **\$accion** tanto en PHP como en los procedimientos almacenados.
- Todas las imágenes que se utilicen deberán estar ubicadas dentro de la carpeta **img** que está en **application/img**; no usar subcarpetas.
- Utilizar la función **Trim()** de php para cualquier *insert*, *update* o *búsqueda*, otra opción es usar **ltrim**, **rtrim** desde Sql Server.
- **Autocompletes**: Al momento de utilizar autocompletes, los campos del *select* en la consulta SQL debe tener los siguientes alias **'id'** y **'value'** y **NO** otros alias; así por ejemplo:

```
SELECT      nProId AS 'id',  
            cProNombre AS 'value'  
FROM        PROCEDIMIENTO
```

- Para recoger el resultado de una operación se utilizarán números que indiquen el estado de la misma:
  - 1 = éxito (texto a mostrar: operación realizada con éxito).
  - 0 = error (texto a mostrar: ha ocurrido un error, vuelva a intentarlo).
  - 2 = vacío (texto a mostrar: no se encontraron registros).
  - 3= existe(texto a mostrar: El elemento buscado ya existe)
- Utilizar “clases” para maquetar una página web, evitar el uso del atributo “style” de las etiquetas html.
- No crear funciones JavaScript en las vistas , optar por crear archivos JavaScript que se referencien desde una vista (html).
- Documentar con algún comentario cada método que se cree en una clase.

Actualizado: 29/08/13

Actualizado: 26/08/13

Actualizado: 25/04/13

**creado:** 16/07/12

Gerencia de Sistemas. Municipalidad Provincial de Trujillo. Todos los derechos reservados ©.