

Instalación

Incluya el archivo:

Función:

```
require_once('path');
```

Parámetros:

path :
Ruta de la clase

Ejemplo de uso:

```
require_once('classes/phpMyDataGrid.class.php');
```

Defina el objeto de la clase:

Función:

```
datagrid('scriptName','gridID');
```

Parámetros:

scriptName :
Nombre del archivo

gridID :
ID del Grid (Se debe usar un código diferente en cada DataGrid si desea tener varios en la misma página)

Ejemplo de uso:

```
$objGrid = new datagrid('sample.php','1');
```

Propiedades

ButtonWidth

Descripción:

Especifica el ancho de los iconos usados como botones en phpMyDataGrid.

Valores posibles:

ButtonWidth :
Número entero.
Predeterminado: 25

Ejemplo de uso:

```
$objGrid -> ButtonWidth = '25';
```

backtick

Descripción:

Algunas bases de datos como MySQL permiten que el nombre de los campos , tablas o bases de datos se encuentren delimitados, por ejemplo con el caracter `

Valores posibles:

backtick :
Caracter.
Predeterminado: `

Ejemplo de uso:

```
$objGrid -> backtick = '`';
```

liquidTable

Descripción:

Define si phpMyDataGrid ocupará automáticamente o no el ancho disponible en pantalla.

Valores posibles:

liquidTable :
Booleano.
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> liquidTable = false;
```

width

Descripción:

Complementa a la propiedad liquidTable definiendo el porcentaje de pantalla que ocupará automáticamente.

Valores posibles:

width :
Cadena, representando un porcentaje.
Predeterminado: 100%

Ejemplo de uso:

```
$objGrid -> width = '100%';
```

condition

Descripción:

Ahora es posible condicionar la presentación de algunos registros para darles otro formato de salida, para ello basta con definir la condición que se debe cumplir.

Valores posibles:

condition :
Cadena, condición.

Ejemplo de uso:

```
$objGrid -> condition = "['activo']==1";
```

IMPORTANTE: Es posible usar cualquier campo que se encuentre en uso dentro del grid, basta con escribir el nombre del campo entre [' ... ']

conditionalStyle

Descripción:

Esta propiedad complementa el uso de la propiedad condition, en esta propiedad se definen los estilos con los que se mostrarán los registros que cumplan la condición especificada.

Valores posibles:

conditionalStyle : style='background:#600;color:#FFF;'
Lista de Estilos CSS

Ejemplo de uso:

```
$objGrid -> conditionalStyle = "style='background:#600;color:#FFF;';"
```

conditionEdit

Descripción:

Si desea limitar la edición de registros a solo los que cumplan con cierta condición, utilice la siguiente propiedad con la condición que se debe cumplir.

Valores posibles:

conditionEdit : ['activo']==1
Cadena, condición.

Ejemplo de uso:

```
$objGrid -> conditionEdit = "['activo']==1";
```

IMPORTANTE: Es posible usar cualquier campo que se encuentre en uso dentro del grid, basta con escribir el nombre del campo entre [' ... ']

conditionDelete

Descripción:

Si desea limitar el borrado de registros a solo los que cumplan con cierta condición, utilice la siguiente propiedad con la condición que se debe cumplir.

Valores posibles:

conditionDelete : ['activo']==1
Cadena, condición.

Ejemplo de uso:

```
$objGrid -> conditionDelete = "['activo']==1";
```

IMPORTANTE: Es posible usar cualquier campo que se encuentre en uso dentro del grid, basta con escribir el nombre del campo entre [' ... ']

moneySign

Descripción:

Define el signo a mostrar en campos de tipo Money

Valores posibles:

moneySign : \$
1 Caracter.
Predeterminado: \$

Ejemplo de uso:

```
$objGrid -> moneySign = '$';
```

zebraLines

Descripción:

Define la forma en que se trazará el interlineado en phpMyDataGrid

Valores posibles:

zebraLines : 1
Entero
Predeterminado: 1

Ejemplo de uso:

```
$objGrid -> zebraLines = '1';
```

showToOf

Descripción:

Esta propiedad activa si se muestra o no el mensaje **Mostrando registros X a Y**

Valores posibles:

showToOf : true
Booleano.
Predeterminado: true

Ejemplo de uso:

```
$objGrid -> showToOf = true;
```

AllowChangeNumRows

Descripción:

phpMyDataGrid permite al usuario cambiar la cantidad de registros que desea visualizar por página, esto puede ser cambiado definiendo esta propiedad como false

Valores posibles:

AllowChangeNumRows : true
Booleano.
Predeterminado: true

Ejemplo de uso:

```
$objGrid -> AllowChangeNumRows = true;
```

sqlDataCoding

Descripción:

Permite definir el tipo de codificación a usar en la base de datos

Valores posibles:

sqlDataCoding : SET NAMES 'utf8'

Ejemplo de uso:

```
$objGrid -> sqlDataCoding = "SET NAMES 'utf8'";
```

charset

Descripción:

Permite definir la codificación de caracteres para mostrar la página.

Valores posibles:

charset : UTF-8

Predeterminado: ISO-8859-1

Ejemplo de uso:

```
$objGrid -> charset = 'UTF-8';
```

defaultdateformat

Descripción:

Permite definir el formato predeterminado para los campos tipo fecha

Valores posibles:

defaultdateformat : ymd

Combinación de las letras dmy

Predeterminado: dmy

Ejemplo de uso:

```
$objGrid -> defaultdateformat = 'ymd';
```

defaultdateseparator

Descripción:

Permite definir el separador para los campos tipo fecha.

Valores posibles:

defaultdateseparator : -

1 Caracter.

Predeterminado: /

Ejemplo de uso:

```
$objGrid -> defaultdateseparator = '-';
```

csvSeparator

Descripción:

Caracter a usar para separar los valores al exportar en formato CSV (Valores separados por coma)

Valores posibles:

csvSeparator : ;
1 caracter
Predeterminado: ;

Ejemplo de uso:

```
$objGrid -> csvSeparator = ',';
```

uploadDirectory

Descripción:

Esta propiedad define el directorio al cual se pueden **subir** o cargar las imágenes

Valores posibles:

uploadDirectory : /images/upload/
ruta de ubicación válida, puede ser relativa o absoluta
Predeterminado: /

Ejemplo de uso:

```
$objGrid -> uploadDirectory = '/images/upload/';
```

IMPORTANTE: La ruta especificada debe tener permisos de escritura.

show404image

Descripción:

Le indica a phpMyDataGrid si debe mostrar o no una imagen representativa cuando no encuentre la imagen relacionada a un campo.

Valores posibles:

show404image : false
Booleano.
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> show404image = false;
```

retcode

Descripción:

De forma predeterminada, phpMyDataGrid es renderizado al terminar la ejecución del script, pero si la propiedad retcode es cambiada a true, se devolverá el código HTML para ser almacenado en una variable.

Valores posibles:

retcode : false
Booleano
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> retcode = false;
```

getMyOwnButtons

Descripción:

Permite al programador tener control sobre la posición de los botones Nuevo/Buscar/Exportar, al no mostrarlos directamente en el grid, sino que permite posicionarlos en cualquier sección de su sitio Web.

Valores posibles:

getMyOwnButtons :
Booleano.
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> getMyOwnButtons = false;
```

IMPORTANTE: Para su funcionamiento, la propiedad **retcode** debe estar puesta a **true**.

strAddBtn:

Función:

Esta propiedad complementa la propiedad **getMyOwnButtons**, devolviendo en ella el código HTML del botón adicionar

Ejemplo de uso:

```
echo $objGrid -> strAddBtn;
```

strSearchBtn:

Función:

Esta propiedad complementa la propiedad **getMyOwnButtons**, devolviendo en ella el código HTML del botón buscar

Ejemplo de uso:

```
echo $objGrid -> strSearchBtn;
```

strExportBtn:

Función:

Esta propiedad complementa la propiedad **getMyOwnButtons**, devolviendo en ella el código HTML del botón exportar

Ejemplo de uso:

```
echo $objGrid -> strExportBtn;
```

cssPrinter

Descripción:

Define el nombre del archivo de estilos que será usado para impresión.

Valores posibles:

cssPrinter :
Ruta y nombre del archivo CSS
Predeterminado: **css/b-w-print.css**

Ejemplo de uso:

```
$objGrid -> cssPrinter = '/css/my_css_file.css';
```

PDFfont

Descripción:

Define el tipo de letra a usar al exportar a PDF

Valores posibles:

PDFfont : Times New Roman
Nombre de tipo de letra
Predeterminado: Arial

Ejemplo de uso:

```
$objGrid -> PDFfont = 'Times New Roman';
```

PDFfontsize

Descripción:

Define el tamaño tipo de letra a usar al exportar a PDF

Valores posibles:

PDFfontsize : 8
Numérico.
Predeterminado: 7

Ejemplo de uso:

```
$objGrid -> PDFfontsize = '8';
```

PDFfill

Descripción:

Define los valores RGB para el color de fondo para los títulos al exportar a PDF

Valores posibles:

PDFfill : array("R"=>0,"G"=>192,"B"=>192)
Predeterminado: array("R"=>192,"G"=>192,"B"=>192);

Ejemplo de uso:

```
$objGrid -> PDFfill = array("R"=>0,"G"=>192,"B"=>192);
```

PDFdraw

Descripción:

Define los valores RGB para el color de la letra al exportar a PDF

Valores posibles:

PDFdraw : array("R"=>0,"G"=>192,"B"=>192)
Predeterminado: array("R"=>0,"G"=>0,"B"=>0);

Ejemplo de uso:

```
$objGrid -> PDFdraw = array("R"=>0,"G"=>192,"B"=>192);
```

actHeader

Descripción:

Se usa para generar un encabezado en las opciones: **Nuevo, Editar, Ver.**

Valores posibles:

actHeader ['add'] : Header for New

```
$objGrid -> actHeader['add'] = 'Header for <strong>New</strong>';
```



<< . >>

actFooter

Descripción:

Se usa para generar un pie de página en las opciones: **Nuevo, Editar, Ver.**

Valores posibles:

actFooter ['add'] : Footer for New

```
$objGrid -> actFooter['add'] = 'Footer for <strong>New</strong>';
```



<< . >>

images

Descripción:

Esta propiedad es un vector que contiene los nombres de los archivos de imagen que se usarán como iconos en el Datagrid.

Valores posibles:

images ['add'] : add.png

```
$objGrid -> images['add'] = 'add.png';
```



IMPORTANTE: Si cambia algún valor de nombre de imagen, asegúrese que la imagen exista en la carpeta de imágenes

<< . >>

message

Descripción:

Esta propiedad es un vector que contiene los textos descriptivos que se usarán en el Datagrid, cambiando sus valores, personalizará el datagrid para que utilice sus propios textos

Valores posibles:

message ['cancel'] : Cancel

```
$objGrid -> message['cancel'] = 'Cancel';
```



<< . >>

debug

Descripción:

Genera mensajes para depurar los resultados de phpMyDataGrid

Valores posibles:

debug : false
Booleano.
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> debug = false;
```

poweredby

Descripción:

Define si el logo de phpMyDataGrid es mostrado

Valores posibles:

poweredby : false
Booleano.
Predeterminado: false

Ejemplo de uso:

```
$objGrid -> poweredby = false;
```

Métodos

Defina el nombre de formulario:

Función:

```
Form('formName', doForm);
```

Parámetros:

formName : employees
Nombre del formulario

doForm : true
true = phpMyDataGrid generará automáticamente el formulario (Debe usar esta opción si desea que phpMyDataGrid cargue imágenes al servidor)
false = phpMyDataGrid usará un formulario creado por el programador

Ejemplo de uso:

```
$objGrid -> Form ('employees', true);
```

Defina el método del formulario:

Función:

```
methodForm('strMethod');
```

Parámetros:

strMethod :

Puede usar cualquiera de los métodos de formulario disponibles, **GET** o **POST**

Ejemplo de uso:

```
$objGrid -> methodForm ('post');
```

IMPORTANTE: Si está haciendo el llamado a phpMyDataGrid desde un formulario definido por el programador, deberá usar el mismo método usado en el formulario.

Si usted traslada a la página del script sus propios parámetros, debería continuar trasladándolos al script del dataGrid, para esto, utilice:

Función:

```
linkparam("parameters");
```

Parámetros:

parameters :

Debe definir la lista de parámetros de la misma forma que se crean parámetros del tipo **GET**

Ejemplo de uso:

```
$objGrid -> linkparam("&session={$session}&userid={$userid}&option=4");
```

IMPORTANTE:
Las variables que se han incluido en la lista de parámetros deben haber sido previamente capturadas, por ejemplo, para capturar la variable **session**:
Si el envío es con GET:
\$session = \$_GET['session'];
Si el envío es con POST:
\$session = \$_POST['session'];
Si no conoce el origen del envío, o el envío puede ser mixto (GET y POST):
\$session = (isset(\$_GET['session'])?\$_GET['session']:(isset(\$_POST['session'])?\$_POST['session']:));
NÓTESE QUE LA LISTA DE PARÁMETROS SE ENCUENTRA DELIMITADA POR COMILLAS DOBLES Y EL NOMBRE DE LAS VARIABLES ENCERRADO ENTRE LLAVES {}, ASÍ LAS VARIABLES SON REEMPLAZADAS POR SU VALOR.

Si desea tener un menú contextual con las opciones de mantenimiento, utilice la siguiente línea:

Función:

```
useRightClickMenu();
```

Ejemplo de uso:

```
$objGrid -> useRightClickMenu();
```

Para definir el modo en que se genera el código HTML:

Función:

```
friendlyHTML(bolStat);
```

Parámetros:

bolStat :
true = si desea que el código HTML generado sea legible
false = Si desea que el código fuente generado esté 'obfusado'

Ejemplo de uso:

```
$objGrid -> friendlyHTML(true);
```

Para compatibilidad con XHTML utilice esta función:

Función:

```
closeTags(bolStat);
```

Parámetros:

bolStat :
true = si desea que el HTML generado sea correcto para la declaración XHTML
false = Si desea que el código generado sea compatible con HTML

Ejemplo de uso:

```
$objGrid -> closeTags(false);
```

Para definir la ruta (camino) en el cual se encuentran los iconos del datagrid:

Función:

```
pathtoimages('strPath');
```

Parámetros:

strPath :
Escriba la ruta en la cual se encuentran ubicados los archivos de imagen

Ejemplo de uso:

```
$objGrid -> pathtoimages('/images/');
```

Es necesario establecer una conexión con el servidor de bases de datos, para esto utilice la siguiente función:

Función:

```
conectadb('strServer', 'strUsername', 'strPassword', 'strDatabase', 'useADODB', 'strType', intPort);
```

Parámetros:

strServer :
Nombre o dirección IP del servidor

strUsername :
Nombre de usuario de la base de datos

strPassword :
Contraseña de usuario de la base de datos

strDatabase :
Nombre de la base de datos

useADODB :
true = Realizará la conexión a la base de datos usando la librería ADODB
false = Utilizará los drivers nativos de php para conectarse a MySQL

strType :
Tipo de servidor de base de datos (aplica solamente para conexiones realizadas con ADOdb)

intPort :
Puerto de escucha de la base de datos

Ejemplo de uso:

```
$objGrid -> conectadb('localhost', 'root', '%my12PaSS%', 'testdb', 'false', 'mysql', 3306);
```

IMPORTANTE: Si decide utilizar la librería ADOdb para realizar la conexión, deberá descargarla desde su sitio web, así mismo, deberá incluirla al inicio del script

El idioma predeterminado para los mensajes de phpMyDataGrid es inglés, si desea cambiar el idioma, utilice esta función:

Función:

```
language('strLang');
```

Parámetros:

strLang :
Escriba el código ISO de dos caracteres del idioma

Ejemplo de uso:

```
$objGrid -> language('es');
```

IMPORTANTE: phpMyDataGrid Professional tiene como idiomas base el español **es** y el inglés **en**, si desea adicionar otro idioma, basta con que lo cree en la carpeta **languages**

Para habilitar o deshabilitar los iconos de mantenimiento, utilice la siguiente función:

Función:

```
buttons(bolAdd, bolUpd, bolDel, bolChk, intColumn, 'strColumnName');
```

Parámetros:

bolAdd :
true = Habilita el sistema de adición de registros del grid
false = Deshabilita el sistema de adición de registros del grid

bolUpd :
true = Habilita el sistema de actualización de registros del grid
false = Deshabilita el sistema de actualización de registros del grid

bolDel :
true = Habilita el sistema de borrado de registros del grid
false = Deshabilita el sistema de borrado de registros del grid

bolChk :
true = Habilita el sistema de visualización de registros del grid
false = Deshabilita el sistema de visualización de registros del grid

intColumn :
Define la posición (columna) en la cual se desea mostrar los iconos de mantenimiento (-1 indica al final del grid)

strColumnName :
Llene este campo, si desea mostrar un título en la columna de iconos

Ejemplo de uso:

```
$objGrid -> buttons(true, true, true, true, -1, 'Column Title');
```

Para habilitar o deshabilitar las opciones de exportar, utilice la siguiente función:

Función:

```
export(bolExportsheet, bolExportCSV, bolExportXML, bolPrinter, bolExportPDF, 'pdfOrientation');
```

Parámetros:

bolExportsheet	:	<input type="checkbox"/>	true = Habilita la opción de exportar a Hoja de cálculo (XLS) false = Deshabilita la opción de exportar a Hoja de cálculo (XLS)
bolExportCSV	:	<input type="checkbox"/>	true = Habilita la opción de exportar a archivo separado por comas (CSV) false = Deshabilita la opción de exportar a archivo separado por comas (CSV)
bolExportXML	:	<input type="checkbox"/>	true = Habilita la opción de exportar a XML false = Deshabilita la opción de exportar a XML
bolPrinter	:	<input type="checkbox"/>	true = Habilita la opción de imprimir false = Deshabilita la opción de imprimir
bolExportPDF	:	<input type="checkbox"/>	true = Habilita la opción de exportar a PDF false = Deshabilita la opción de exportar a PDF
pdfOrientation	:	<input type="text" value="P"/>	Selecciona la orientación de la página al exportar a PDF (P) = Vertical (L) = Horizontal

Ejemplo de uso:

```
$objGrid -> export(true, true, true, true, true, 'P');
```

Si desea tener una columna con casillas de verificación para selección múltiple, utilice la siguiente función:

Función:

```
checkable(status);
```

Parámetros:

status	:	<input type="checkbox"/>	true = Habilita la columna de casillas de verificación false = Deshabilita la columna de casillas de verificación
--------	---	--------------------------	--

Ejemplo de uso:

```
$objGrid -> checkable(false);
```

Para definir un título en el grid, utilice:

Función:

```
TituloGrid('strTitle');
```

Parámetros:

strTitle	:	<input type="text" value="Grid Title"/>
----------	---	---

Ejemplo de uso:

```
$objGrid -> TituloGrid('Grid Title');
```

Para definir un pie de página en el grid, utilice:

Función:

```
FooterGrid('strFooter');
```

Parámetros:

strFooter :

Ejemplo de uso:

```
$objGrid -> FooterGrid('Grid footnote, © 2008 Gurú Sistemas');
```

Para controlar la cantidad de registros a visualizar por cada página:

Función:

```
datarows(intLines);
```

Parámetros:

intLines :
Cantidad de registros por página

Ejemplo de uso:

```
$objGrid -> datarows(15);
```

Si desea definir la cantidad de páginas que se mostrarán antes de resumir con ... puede hacerlo cambiando la cantidad con esta función:

Función:

```
linksperpage('amount');
```

Parámetros:

amount :
Escriba la cantidad de links que desea visualizar

Ejemplo de uso:

```
$objGrid -> linksperpage('4');
```

IMPORTANTE: No defina cantidades muy grandes ya que pueden distorsionar la estructura del Grid

Si define linksperpage = 5, verá algo como:

1 2 3 4 5 ... 15 16 17 18 19 **20** 21 22 23 24 25 ... 45 46 47 48 49

Para cambiar el formato de paginación, utilice:

Función:

```
paginationmode('pgm',inTable);
```

Parámetros:

pgm :

Defina el tipo de paginación, existen 3 valores disponibles:

links = Genera un a lista de números de página que indican el número de página al que se desea ir (Recomendado tablas con no mas de 20 páginas)

select = Genera un menú desplegable que permite elegir el número de página a la que desea ir

mixed = Genera un listado combinando los métodos links y select (valor predeterminado)

inTable :

true = Los números de página se formatearán dentro de una tabla

false = Los números de página se presentarán normalmente

Ejemplo de uso:

```
$objGrid -> paginationmode('mixed', false);
```

Puede definir un código de seguridad (palabra mágica) que ayudará a phpMyDataGrid a ser más seguro:

Función:

```
salt('code');
```

Parámetros:

code :

Ejemplo de uso:

```
$objGrid -> salt('salt&pepper');
```

Defina la cantidad de dígitos decimales que desea mostrar en los campos de tipo numérico:

Función:

```
decimalDigits('amount');
```

Parámetros:

amount :

Ejemplo de uso:

```
$objGrid -> decimalDigits('2');
```

Puede definir el caracter que desea utilizar como separador decimal:

Función:

```
decimalPoint('char');
```

Parámetros:

char :

Ejemplo de uso:

```
$objGrid -> decimalPoint('.');
```

La edición en línea puede estar: desactivada, activarse con un solo click y grabar al salir del campo, o solicitar confirmación de grabación, esto puede ser definido desde la función:

Función:

```
ajax('style');
```

Parámetros:

style	:	<input type="text" value="none"/>
-------	---	-----------------------------------

none = Deshabilita la edición en línea
default = Habilita la edición en línea con confirmación de grabación
silent = Habilita la edición en línea con grabación automática

Ejemplo de uso:

```
$objGrid -> ajax('none');
```

Si desea diferenciar en pantalla los valores que han sido modificados via AJAX, puede usar esta función:

Función:

```
AjaxChanged('strColor');
```

Parámetros:

strColor	:	<input type="text" value="#900"/>
----------	---	-----------------------------------

Debe ser un color hexadecimal válido

Ejemplo de uso:

```
$objGrid -> AjaxChanged('#900');
```

Si por algún motivo requiere controlar si se está realizando una llamada AJAX a la página puede usar esta función:

Función:

```
isAjaxRequest();
```

Ejemplo de uso:

```
$objGrid -> isAjaxRequest();
```

IMPORTANTE: Puede usar la función para determinar si la solicitud a la página fue una llamada AJAX

```
if ($objGrid -> isAjaxRequest()){
  echo 'esto se hace en el proceso AJAX';
} else {
  echo 'esta es una solicitud directa';
}
```

Usted puede personalizar las acciones que realizarán los botones de mantenimiento:

Función:

```
setAction('button', 'event');
```

Parámetros:

button	:	<input type="text" value="add"/>
event	:	<input type="text" value="javascript_function()"/>

Ejemplo de uso:

```
$objGrid -> setAction('add', 'javascript_function()');
```

IMPORTANTE: Consideraciones a tener en cuenta a la hora de asignar nuevas acciones a los botones:

Botón Adicionar = Ninguna en especial

Ejemplo: \$objGrid -> setAction('add','nuevo_adicionar());

Botón Editar = la función deberá contener dos parámetros, los cuales deben ir delimitados así: ("%s","%s");

Ejemplo: \$objGrid -> setAction('edit','nuevo_editrow("%s","%s");

Botón Borrar = la función deberá contener dos parámetros, los cuales deben ir delimitados así: ("%s","%s");

Ejemplo: \$objGrid -> setAction('delete','nuevo_deleterow("%s","%s");

Botón Buscar = Ninguna en especial

Ejemplo: \$objGrid -> setAction('search','nuevo_buscar());

Botón Ver = la función deberá contener dos parámetros, los cuales deben ir delimitados así: ("%s","%s");

Ejemplo: \$objGrid -> setAction('view','nuevo_viewrow("%s","%s");

Es necesario definir la tabla sobre la cual trabajará el grid:

Función:

```
tabla('strTable');
```

Parámetros:

```
strTable : employees
```

Ejemplo de uso:

```
$objGrid -> tabla('employees');
```

Puede definir una condición del tipo WHERE para filtrar y mostrar solo los registros que cumplan con una condición:

Función:

```
where('strWhere');
```

Parámetros:

```
strWhere : active = 1
```

Ejemplo de uso:

```
$objGrid -> where('active = 1');
```

Si necesita agrupar registros por algún campo, o campos, puede usar la función:

Función:

```
groupby('strGroup');
```

Parámetros:

```
strGroup : status
```

Ejemplo de uso:

```
$objGrid -> groupby('status');
```

IMPORTANTE: Tenga en cuenta que al agrupar datos, las funcionalidades de mantenimiento (Adicionar, Editar, Borrar, edición en línea), no funcionarán correctamente, por lo tanto se recomienda deshabilitarlas, o si es necesario, aplicar sus propios procesos de mantenimiento.

Defina los campos por los cuales desea ordenar los registros:

Función:

```
orderby('fields','style');
```

Parámetros:

fields :

Lista de campos por los que desea ordenar la salida, separada por comas

style :

Defina el tipo de ordenamiento para cada campo **ASC** o **DESC**, en caso de dejar en blanco, se usará automáticamente **ASC**

Ejemplo de uso:

```
$objGrid -> orderby('name,lastname','asc,desc');
```

La versión professional cuenta con una característica de ordenamiento manual, la cual sirve, por ejemplo para definir el orden en el que aparecerán los productos de una página:

Función:

```
setorderarrows('field');
```

Parámetros:

field :

El campo de consecutivo NO debería ser el ID

Ejemplo de uso:

```
$objGrid -> setorderarrows('consecutivo');
```

IMPORTANTE: El campo de ordenamiento debe ser consecutivo y no debe ser autoincrementable, para los ejemplos prácticos que hemos desarrollado, siempre el consecutivo es inicialmente igual al ID autoincrementable del registro.

Por defecto, phpMyDataGrid habilita las flechas de ordenamiento en los títulos de todas las columnas, si desea desactivar esta característica utilice la siguiente función:

Función:

```
noorderarrows();
```

Ejemplo de uso:

```
$objGrid -> noorderarrows();
```

IMPORTANTE: Si desea deshabilitar el ordenamiento de solo unas cuantas columnas del grid, NO utilice esta función, en su lugar utilice **chField** especificando los parámetros necesarios.

phpMyDataGrid Genera automáticamente las consultas SQL basado en la información suministrada, pero en algunas ocasiones es necesario generar consultas avanzadas que requieren que el programador las defina manualmente, para eso puede usar esta función:

Función:

```
sqlstatement('strSQL','strCount');
```

Parámetros:

```
strSQL      : SELECT *, now() as fecha from employees  
strCount    : select count(*) from employees
```

Ejemplo de uso:

```
$objGrid -> sqlstatement('SELECT *, now() as fecha from employees','select count(*) from employees');
```

IMPORTANTE: Las consultas SQL manuales no deben incluir las sentencias WHERE, GROUP u ORDER, estas deben ser definidas directamente desde las funciones disponibles.
También debe tener en cuenta que es muy importante que todos los nombres de campos que utilice en el grid, se encuentren definidos en la consulta SQL

Para el programador es indispensable poder hacer seguimiento al comportamiento de los scripts, esta función permite al programador recibir correos electrónicos con 'eventuales' errores SQL que se puedan generar:

Función:

```
reportSQLErrorsTo('strMail', 'bolShow');
```

Parámetros:

```
strMail      : developer@thecompanymail.com  
              Escribe la dirección de e-mail del programador  
bolShow      : false  
              true = Muestra los errores SQL en pantalla durante la ejecución (Recomendado en desarrollo)  
              false = Oculta cualquier error SQL generado (Recomendado en entornos productivos)
```

Ejemplo de uso:

```
$objGrid -> reportSQLErrorsTo('developer@thecompanymail.com', 'false');
```

IMPORTANTE: Esta función hace uso de la sentencia **mail()** de php, por lo tanto para que funcione correctamente, deberá estar configurada y activa

Para operaciones de mantenimiento es necesario definir un campo clave, utilice la función:

Función:

```
keyfield('strField');
```

Parámetros:

```
strField     : id
```

Ejemplo de uso:

```
$objGrid -> keyfield('id');
```

phpMyDataGrid le permite hacer búsquedas, puede definir los campos por los que desea buscar información usando la siguiente función:

Función:

```
searchby('listoffields');
```

Parámetros:

listoffields :
Puede adicionar la instrucción **:SELECT** al nombre del campo para mostrar un menú desplegable con los posibles valores de búsqueda.

Ejemplo de uso:

```
$objGrid -> searchby('firstname,lastname,date:select');
```

Si desea generar un enlace para controlar el proceso de 'desfiltrar' después de realizar búsquedas, puede usar la función:

Función:

```
getResetSearch();
```

Ejemplo de uso:

```
$objGrid -> getResetSearch();
```

Una de las funciones principales es FormatColumn, con esta función usted definirá las características de cada uno de los campos que mostrará en grid:

Función:

```
FormatColumn('strfieldName','strHeader','fieldWidth','maxlength','inputtype','columnwidth','align','Mask','default','cutChar');
```

Parámetros:

strfieldName :
Nombre del campo en la tabla

strHeader :
Título de la columna

fieldWidth :
Solo usado en campos de tipo **textarea**, identifica la cantidad de líneas que contendrá el textarea

maxlength :
Longitud máxima de caracteres a aceptar en el campo

inputtype :
Tipo de campo
0 = Campo Normal
1 = Campo de solo lectura
2 = Campo Oculto
3 = Imágen, cálculo o enlace sin relación con campo en la tabla
4 = Imágen, cálculo o enlace relacionado con un campo en la tabla

columnwidth :
Ancho de la columna (En píxeles)

align :
Alineación del texto en la columna
center = Centrado (Valor predeterminado)
left = Ajuste del texto a la Izquierda
right = Ajuste del campo a la derecha

Mask :
Enmascaramiento para el campo
text = Campo normal de texto (Valor predeterminado)
textarea = Región de edición de texto (puede tener un mayor área que los campos tipo **text**)

image = Muestra una imagen, puede ser relacionada con un campo o fija. (ver ejemplos)
 imagelink = Muestra una imagen con enlace, puede ser relacionada a un campo o fija. (ver ejemplos)
 number = Campo numérico
 money = Campo numérico con formato de moneda, forma de uso: money:signo, ejemplo **money:\$ money:£**
 date = Campo tipo fecha, forma de uso: date:formato:separador, ejemplo **date:dmy:/ date:yymd-**
 link = Campo con enlace. ver ejemplos
 password = Campo tipo contraseña (Protegido con asteriscos)
 calc = Campo calculado. ver ejemplos
 scalc = Campo calculado que almacena el valor del cálculo, ver ejemplos
 bool = Campo Booleano, genera una casilla de verificación y almacena 0 si no esta chequeada y 1 si esta chequeada
 check = Igual al campo tipo **bool**
 select = Campo con menú de opciones, las opciones pueden ser manualmente definidas, o dinámicamente desde otra tabla de la base de datos.
 0 = Campo numérico sin decimales
 1 = Campo numérico con 1 decimales
 2 = Campo numérico con 2 decimales
 3 = Campo numérico con 3 decimales
 4 = Campo numérico con 4 decimales
 integer = Campo numérico sin decimales

default :
 Valor predeterminado del campo (se usa solo en la opción de adicionar nuevos registros)

cutChar :
 Util en campos de tipo **textarea** que contengan mucha información, con esta opción solo mostrará los primeros **X** caracteres mientras visualice la información en el grid, para ver la información completa puede usar la opción **Ver registro**

Ejemplo de uso:

```
$objGrid -> FormatColumn('lastname', 'Lastname', '0', '20', '0', '80', 'center', 'text', '', '');
```

Puede definir características adicionales a cada una de las columnas, para esto basta con usar:

Función:

chField('strfieldName', 'permissions')

Parámetros:

strfieldName :
 Nombre del campo
 permissions :
 N+ = Visualizar campo al adicionar registro
 N- = Ocultar campo al adicionar registro
 E+ = Visualizar campo al editar registros
 E- = Ocultar campo al editar registros
 V+ = Visualizar campo al ver registro
 V- = Ocultar campo al ver registro
 R = Quitar flechas de ordenamiento a este campo
 U = Si es un campo de tipo **image** permita cargar fotos en el campo
 M = Habilita la opción de cargar imágenes sobre imágenes ya existentes

Ejemplo de uso:

```
$objGrid -> chField('nombrecampo', 'N+')
```

IMPORTANTE: Tenga en cuenta que si desea asignar varios modificadores al mismo campo, debe hacerlo en la misma solicitud, por ejemplo:
 \$objGrid -> chField('firstname', 'N-E+V+R');

Si desea realizar validaciones javascript sobre el campo, utilice la siguiente función:

Función:

jsValidate('strField', 'strValidation', 'strErrorMessage', 'strDisplayMessage');

Parámetros:

strField	:	firstname	Nombre del campo
strValidation	:	this.value.length>=8	Código JavaScript para validar el campo, (Puede tambien llamar una función JS)
strErrorMessage	:	Name must have at least 7 chars long	Mensaje de error cuando no se cumpla la condición
strDisplayMessage	:	Write customer name	Mensaje descriptivo de la información que deberá digitar el usuario

Ejemplo de uso:

```
$objGrid -> jsValidate('firstname', 'this.value.length>=8', 'Name must have at least 7 chars long', 'Write customer name');
```

IMPORTANTE: La validación Javascript se usará en los procesos **Nuevo** y **Edición**, y también en el proceso de edición en línea, cabe tener en cuenta que las validaciones solo se realizan sobre el campo activo, no se pueden realizar operaciones con los otros campos de la tabla.

Si no desea validar la entrada de datos, pero desea dar indicaciones al usuario sobre la información que deberá digitar, utilice:

Función:

```
fldComment('strField', 'strDisplayMessage');
```

Parámetros:

strField	:	lastname	Nombre del campo
strDisplayMessage	:	Write customer lastname	Mensaje de ayuda al usuario

Ejemplo de uso:

```
$objGrid -> fldComment('lastname', 'Write customer lastname');
```

IMPORTANTE: Los mensajes de ayuda no serán mostrados durante el proceso de edición en línea, sin embargo, se visualizarán en los procesos normales de crear y editar registros.

Si en su lista de campos existe alguno de tipo "date" y desea que phpMyDataGrid utilice un calendario (datepicker) para la selección de fechas, utilice:

Función:

```
useCalendar(bolCalendar);
```

Parámetros:

bolCalendar	:	true
-------------	---	------

Ejemplo de uso:

```
$objGrid -> useCalendar(true);
```

Para totalizar columnas, utilice:

Función:

```
total('fields');
```

Parámetros:

fields : salary, days, total
Lista de campos a totalizar separados por coma

Ejemplo de uso:

```
$objGrid -> total('salary, days, total');
```

IMPORTANTE: Recuerde que ahora también puede totalizar columnas calculadas

Por compatibilidad con versiones anteriores existe una función que crea los llamados a los archivos JS y CSS, si desea usarla, utilice:

Función:

```
setHeader('phpScriptFile', 'jsFile', 'cssFile', 'jsCalFile', 'cssCalFile', 'jsmenu');
```

Parámetros:

phpScriptFile : sample.php
Nombre del script que se esta ejecutando

jsFile : js/dgscripts.js
Ruta y nombre del archivo de scripts del datagrid

cssFile : css/dgstyle.css
Ruta y nombre del archivo de estilos del datagrid

jsCalFile : js/dgcalendar.js
Ruta y nombre del archivo que contiene el calendario Datepicker

cssCalFile : css/dgcalendar.css
Ruta y nombre del archivo de estilos del calendario

jsmenu : mmscripts.js
Ruta y nombre del archivo que controla el menú contextual en el grid

Ejemplo de uso:

```
$objGrid -> setHeader('sample.php', 'js/dgscripts.js', 'css/dgstyle.css',  
'js/dgcalendar.js', 'css/dgcalendar.css', 'mmscripts.js');
```

IMPORTANTE: Para mayor flexibilidad y facilidad del programador se recomienda NO usar esta función, en su lugar inserte los archivos CSS y JS como normalmente lo haria con otros archivos de este tipo.

Finalmente, realice el llamado a la función que renderizará el grid basandose en toda la configuración anteriormente suministrada:

Función:

```
grid();
```

Ejemplo de uso:

```
$objGrid -> grid();
```