

# PHP



**Realizado por el Ingeniero Francisco Riccio.**

# Temas

- PHP inicial (Tipos de datos, conversiones, etc).
- Clases y objetos.
- Manejo de archivos.
- Base de datos.
- Manejo de cookies y sesiones.
- Web Services.
- Temas variados (COM, Java, PDF, gráficos, expresiones regulares, manejo de directorios).

# PHP Inicial - Tipos de datos

- Integer.
- Double.
- Boolean.
- String.
- Array.
- Object.

# PHP Inicial - Conversiones

- Cambiar el tipo de dato a una variable:  
`settype($var, "tipo de dato")`.
- Retornar el tipo de dato de una variable:  
`gettype($var)`.
- Casting, ejemplo: `(tipo de dato) $var`.
- Funciones relacionadas: `is_bool($var)`,  
`is_int($var)`, `is_long($var)`, `is_double($var)`,  
`is_string($var)`, `is_object($var)`, etc.

# PHP Inicial - Uso de constantes.

Para crear constantes se realiza de la siguiente manera: `define("nombre",valor)`, se llama sin el uso de `$`.

El manejo de excepciones se realiza mediante la función: `error_reporting(constante)`.

Las constantes son: `E_ERROR` (Error crítico sin recuperación posible), `E_WARNING` (Condición de error que no impide que se continúe interpretado), `E_PARSE` (Error de sintaxis), `E_NOTICE` (Evento que no detiene la ejecución, ejemplo: variable no inicializada. Y `E_ALL` (Conjunto de todos los errores).

Normalmente se pone al comienzo de las páginas php.

Otras constantes son: `TRUE`, `FALSE` y `NULL`.

# PHP Inicial - Manipulación de cadenas.

- `chr(#)`.- Devuelve el carácter ascii asociado.
- `ord(caracter)`.- Devuelve el número ascii asociado a un carácter.
- `strtolower($var)` y `strtoupper($var)`.- Convierte a minúscula y mayúscula respectivamente.
- `strlen($var)`.- Devuelve la cantidad de caracteres.
- `$var1.$var2`.- Concatena.
- `strpos($var,$subcadena)`.- Devuelve la posición de la subcadena, es false si no la encuentra.

# PHP Inicial - Manipulación de fechas.

- `time()`.- Devuelve un long con la cantidad de segundos desde el 01-01-1979 hasta la actualidad.
- `getdate(long_fecha)`.- Devuelve un vector con los siguientes campos: seconds, minutes, hours, mday, mon, year y otros menos importantes.
- `mktime(hora,minutos,segundos,mes,día,año)`.- Devuelve un long con los valores enviados.
- `checkdate(mes,día,año)`.- Devuelve true si la fecha ingresada es válida.

# PHP Inicial - Vectores y Matrices.

- Para crear un vector basta con usar corchetes. Ejemplo: `var[1000]`.
- Las matrices se utiliza doble corchete. Ejemplo: `var[25][30]`.
- En vez de utilizar índices se puede utilizar nombres. Ejemplo: `var['nombre']`.

# PHP Inicial - Vectores y Matrices (complemento).

Funciones relacionadas:

- `count(vector)` o `sizeof(vector)`.- Devuelve el número de elementos.
- `next`, `reset`, `prev`, `current` y `end`.- Se desplaza por el vector.
- `array_splice (vector, pos ini, tamaño)`.- Elimina la posición de un elemento.

# PHP Inicial - Variables de HTML.

Existen 2 maneras de obtener el valor de las variables de una página HTML:

1. Se debe configurar el archivo PHP.ini y la variable `register_globals` activarla, luego con se puede referenciar en php de esta manera:  
`$varHTML`.
2. Usando los arreglos:  
`$HTTP_GET_VARS["varHTML"]`,  
`$HTTP_POST_VARS["varHTML"]` y  
`$HTTP_POST_FILES["varHTML"]`.

# PHP Inicial - Variables de HTML (complemento).

- `isset($var)`.- Devuelve true si la variable existe.
  - `empty($var)`.- Devuelve true si la variable está definida pero no se le ha asignado ningún valor.
  - `unset($var)`.- Libera los recursos asociados a las variables que se le pasan como parámetros. Devuelve 1 si no ha habido error y 0 si lo hubo.
- \*. Para redirigir el script a una página, se puede usar el siguiente comando: `header("Location: ruta web"); exit()`.
- \*. Podemos cambiar el content type de la siguiente manera: `header("Content-type: content type")`.

# PHP Inicial - Recepción de un archivo.

Los pasos son los siguientes:

- **Lado del cliente.-** El formulario debe tener la propiedad ENCTYPE='multipart/form-data', agregar un hidden con el tamaño máximo del file, ejemplo:  
`<input type="hidden" name="MAX_FILE_SIZE" value="tamaño máximo en byes ejemplo: 102400">` y un componente del tipo file.

# PHP Inicial - Recepción de un archivo.

- **Lado del servidor.-** Las propiedades para recoger la variable file son las siguientes:  
`$HTTP_POST_FILES["varHTML"]["propiedad"]`,  
donde propiedad puede ser: type, name, size y tmp\_name (ruta temporal donde se almacena el file).

# PHP Inicial - Funciones.

- Función con parámetro por valor: function nombre (\$par).
- Función con parámetro por referencia: function nombre (&\$par).
- Función con parámetro con valor por defecto: function nombre (\$par = valor).

Nota:

- Para retornar el valor de una función se usa return.
- El PHP permite recursividad en sus funciones.

# PHP Inicial - Funciones (complemento).

- Variables del tipo static (static \$var) siempre retorna el valor último que tuvo esa variable.
- Variables del tipo global (global \$var) son globales en toda la aplicación.
- Para conseguir una función con parámetros n se utiliza la función `func_nums_args()` que devuelve el número de parámetros enviados y con `func_get_arg(i)` se obtiene el parametro i enviado.

# PHP Inicial - Inclusión de archivos.

- `require()` .- Une el código del archivo llamado en el archivo actual, la unión es siempre obligatoria. Ejemplo: `require("pag.php")`.
- `include()`.- Tiene la misma función del `require` pero la unión no es obligatoria, la podemos condicionar con algunos `if`, por ejemplo: `if ($var = valor) include("pag.php.")`.
- `require_once()` o `include_once()`.- Permite que no exista problemas al leer 2 o más veces un archivo. La diferencia entre ambos es la misma que `include()` y `require()`.

# PHP Inicial - Principales Funciones predeterminadas

- `phpinfo()`.- Muestra la configuración del PHP en el servidor.
  - `extension_loaded("librería")`.- Devuelve true si la librería esta cargada.  
Ejemplo: `extension_loaded("mysql")`.
  - `$_SERVER["SERVER_NAME"]`.- Nombre del servidor.
  - `$_SERVER["SCRIPT_NAME"]`.- Nombre del script.
  - `$REMOTE_ADDR`.- IP del cliente.
  - `getenv(constante)`.- Devuelve cualquier constante del servidor.  
`$HTTP_USER_AGENT`.- Navegador de Internet del cliente.
  - `$HTTP_ACCEPT`.- Content types permitidos por el navegador.
  - `$HTTP_ACCEPT_LANGUAGE`.- Idioma del navegador.
- \* Las constantes se pueden conseguir de la función `phpinfo()`.

# Clases y objetos.

Estructura:

```
class ClassName  
{  
  
}
```

Llamada:

- `$obj = new class() luego $obj->función().`
- `Clase::función.`

# Clases y objetos.

## Observaciones:

- El comando `extends` permite la herencia de clases.
- Los atributos de las clases se declaran de la siguiente manera: `var $atributo1`.
- La llamada al mismo objeto es mediante `$this->`.
- La llamada a la clase padre es mediante el comando `parent`.
- El constructor de una clase hija por defecto no llama al constructor de la clase padre.

# Clases y objetos - Reflexión.

- Conseguir las funciones de un objeto:  

```
$m = get_class_methods(get_class($obj));  
foreach ($m as $method)  
    $method //Nombre del método en string.
```
- Conseguir los atributos de un objeto:  

```
$a = get_object_vars($obj);  
while (list($prop, $val) = each($a))  
    $val //Nombre de cada atributo en string.
```

# Clases y objetos - Reflexión.

- `is_subclass_of($$obj, $class)`.- Devuelve true si el objeto pertenece a dicha clase.
- `get_parent_class($obj)`.- Devuelve la clase padre del objeto.
- `get_class($obj)`.- Devuelve la clase del objeto.
- Llamada de un método por reflexión:  
`error_reporting(E_WARNING);`  
`call_user_method ("método", $obj,`  
`"par1", "par2", ..., "parn");`

# Clases y objetos – Serialización.

Ejemplo:

```
$obj1 = new clase1;  
$sobj1 = serialize($a); //Serializa.  
$fp = fopen("archivo", "w");  
fputs($fp, $sobj1);  
fclose($fp);  
$sobj2 = implode("", @file("archivo"));  
$obj2 = unserialize($sobj2); //Deserializa.  
Donde al finalizar: $obj1 = $obj2
```

# Manejo de archivos - Planos.

Abrir y cerrar un archivo:

```
$file = fopen("archivo", "opc");  
fclose($file)
```

Donde opc es:

r.- Sólo lectura, r+.- Lectura y escritura.

w.- Sólo lectura pero borra el archivo, w+.-  
Lectura y escritura pero borra el archivo.

a.- Sólo lectura y conserva el contenido del  
archivo, a+.- Lectura y escritura conservando el  
contenido del archivo.

# Manejo de archivos - Planos (funciones).

- Lectura:

`fgetc($file)`.- Devuelve un carácter.

`fgets($file, n)`.- Devuelve n-1 bytes.

`fread($file, n)`.- Devuelve n bytes.

`fscanf($file, "%p1..%pn", $var,..,$varn)`.-

Devuelve una conjunto de bytes del archivo con la estructura configurada.

P puede ser:

`%d` = número, `%s` = cadena, `\t` = tabulación y `\n` = fin de línea.

# Manejo de archivos - Planos (funciones).

- Escribir:

`fwrite($file, "texto", longitud de bytes).`

- Posicionamiento:

`fseek($file, # byte).`- Se posiciona sobre el byte que se desea, devuelve 0 si lo logro.

`ftell($file).`- Devuelve el byte donde se encuentra posicionado el cursor.

# Manejo de archivos - XML.

- Crear un objeto de la clase XmlDocument:

```
$dom = new XmlDocument();
```

- Llamar el archivo xml:

```
$dom->load("archivo.xml");
```

- Operaciones:

getElementsByTagName.- Devuelve un tag del xml.

Ejemplo:

```
$params = $dom->getElementsByTagName("tag");
```

```
foreach ($params as $param) {
```

```
    echo $param -> getAttribute('name');
```

```
    echo $param -> getAttribute('value'); }
```

# Manejo de archivos - XML (complemento).

createElement.- Crea un elemento en el archivo XML.

Ejemplo:

```
$e = $doc->createElement("tag", "valor");
```

```
$doc->appendChild($e);
```

```
$doc->saveXML();
```

schemaValidate('archivo').- Devuelve true si el esquema es válido.

Ejemplo:

```
if (!$dom->schemaValidate("archivo.xsd"))
```

```
print "Documento invalido";
```

# Manejo de archivos - XML (complemento).

save().- Graba un objeto dom en un archivo xml.

Ejemplo:

```
$dom->save("archivo.xml").
```

\*. Para trabajar con XML sobre la versión de PHP 4.3, debemos copiar el archivo incov.dll en la carpeta de Windows y habilitarle la opción en el archivo php.ini.

# Base de datos - ODBC.

- `odbc_connect`.- Crea una conexión.  
Ejemplo: `$con = odbc_connect("dns", "user", "pass")`.
- `odbc_pconnect`.- Crea una conexión reusable, no se desconecta. Si un usuario tiene el mismo usuario y password se le entrega esta conexión.  
Ejemplo: `$con = odbc_pconnect("dns", "user", "pass")`.
- `odbc_close`.- Cierra una conexión.  
Ejemplo: `odbc_close($con)`.

Nota: Con ODBC hay que tener cuidado con la función `odbc_num_rows` porque devuelve `-1` con algunas bases de datos.

# Base de datos - ODBC (transacciones).

- `odbc_autocommit`.- Establece una transacción. Se recomienda que el segundo parámetro sea `false`.

Ejemplo: `odbc_autocommit($con, FALSE);`

`odbc_do($con, $sql); //n veces.`

`odbc_commit($con);`

Nota: Si se sigue haciendo más transacciones sobre la conexión y el parámetro es `FALSE`, no se efectuará las transacciones hasta que se llame a la función `odbc_commit`.

- `odbc_rollback`.- Realiza rollback sobre transacciones.

Ejemplo: `odbc_rollback($con);`

# Base de datos - ODBC (operaciones).

- `odbc_do`.- Ejecuta un query.

Ejemplo: `$rs = odbc_do($con, $sql)`.

- `odbc_prepare`.- Prepara un query para luego ejecutarlo con `odbc_execute`.

`$stmt = odbc_prepare($con, $qry)`.

`$rs = odbc_execute($stmt)`.

Nota: `odbc_execute` devuelve true si no tuvo problemas para ejecutar el comando sql.

# Base de datos - ODBC (comandos).

- `odbc_free_result`.- Libera un recurso utilizado para las consultas.

Ejemplo: `odbc_free_result($rs)` o  
`odbc_free_result($stmt)`.

- `odbc_num_fields`.- Devuelve el número de columnas traídas.
- `odbc_result`.- Devuelve el valor de una fila.

Ejemplo:

`odbc_result($rs, n);` //Donde n comienza en 1.

- `odbc_fetch_row`.- Devuelve true si el cursor aún no llega al final.

# Base de datos – MySQL (creación de tablas).

Script para MySQL para crear tablas relacionadas:

```
CREATE TABLE tabla1 (  
  `campo1` tipo de dato NOT NULL default valor,  
  `campo2` tipo de dato default NULL,  
  PRIMARY KEY (`campo1,...,campon`))  
TYPE=InnoDB;
```

# Base de datos - MySQL (creación de tablas).

```
CREATE TABLE tabla2
```

```
(
```

```
  `campo1` tipo de dato NOT NULL,
```

```
  `campo2` tipo de dato NOT NULL,
```

```
  PRIMARY KEY(campo1),
```

```
  INDEX (campo2),
```

```
  FOREIGN KEY (campo2) REFERENCES
```

```
  tabla1(campo1)
```

```
) TYPE = INNODB;
```

# Base de datos - MySQL (transacciones).

MySQL por defecto tiene `autocommit(true)` y para cambiarlo se debe ingresar cualquiera de los siguientes comandos:

- `SET AUTOCOMMIT = 0.`
- `BEGIN.`

Al finalizar la transacción debe ir el comando `COMMIT` para asegurar que se grabe los cambios en la base de datos y con el comando `ROLLBACK` se realiza lo contrario.

Ejemplo: `begin; operación 1...operación n; commit.`

# Base de datos - MySQL.

- `mysql_connect`.- Crea una conexión.

Ejemplo: `$con = mysql_connect($host, $user, $password)`.

- `mysql_pconnect`.- Crea una conexión reutilizable, no se desconecta. Si un usuario tiene el mismo usuario y password se le entrega esta conexión.

Ejemplo:

`$con = mysql_pconnect($host, $user, $password)`.

- `mysql_select_db`.- Selecciona una base de datos.

Ejemplo: `mysql_select_db("bd", $con)`;

- `mysql_close`.- Cierra una conexión.

Ejemplo: `mysql_close($con)`.

# Base de datos - MySQL

## (operaciones).

- `mysql_query`.- Ejecuta un query.  
Ejemplo: `mysql_query($qry, $con)`.
- `mysql_affected_rows`.- Devuelve el número de registros afectados en una transacción.  
Ejemplo: `mysql_affected_rows($rs)`.
- `mysql_num_rows`.- Devuelve el número de registros obtenidos.  
Ejemplo: `mysql_num_rows($rs)`.
- `mysql_result`.- Devuelve el valor de un campo de un registro.  
Ejemplo: `mysql_result($rs, $fila, "campo")`.

# Base de datos - MySQL (operaciones).

- `mysql_fetch_row`.- Devuelve una fila.  
Ejemplo: `$fila = mysql_fetch_row($rs)`.
- `mysql_num_fields`.- Devuelve el número de campos.  
Ejemplo: `mysql_num_fields($rs)`.
- `mysql_list_dbs`.- Devuelve la lista de base de datos en el servidor.  
Ejemplo: `mysql_list_dbs($con)`.
- `mysql_get_server_info()`.- Devuelve la versión del servidor.

# Base de datos - MySQL (operaciones).

- `mysql_error()`.- Devuelve el error conseguido.
- `mysql_free_result()`.- Libera un recurso utilizado para las consultas.

Ejemplo: `mysql_free_result($rs)` o  
`mysql_free_result($stmt)`.

# Base de datos - SQL Server

- `mssql_connect`.- Crea una conexión.

Ejemplo:

```
$con = mssql_connect($host, $user, $password).
```

- `mssql_pconnect`.- Crea una conexión reutilizable, no se desconecta. Si un usuario tiene el mismo usuario y password se le entrega esta conexión.

Ejemplo:

```
$con = mssql_pconnect($host,$user,$password).
```

- `mssql_select_db`.- Selecciona una base de datos.

Ejemplo: `mssql_select_db("bd", $con);`

- `mssql_close`.- Cierra una conexión.

Ejemplo: `mssql_close($con).`

# Base de datos - SQL Server (operaciones).

- `mssql_query`.- Ejecuta un query.  
Ejemplo: `mssql_query($qry,$con)`.
- `mssql_num_rows`.- Devuelve el número de registros obtenidos.  
Ejemplo: `mssql_num_rows($rs)`.
- `mssql_result`.- Devuelve el valor de un campo de un registro.  
Ejemplo: `mssql_result($rs, $fila, "campo")`.

# Base de datos - SQL Server (operaciones).

- `mssql_fetch_row`.- Devuelve una fila.  
Ejemplo: `$fila = mssql_fetch_row($rs)`.
- `mssql_num_fields`.- Devuelve el número de campos.  
Ejemplo: `mssql_num_fields($rs)`.
- `mssql_free_result`.- Libera un recurso utilizado para las consultas.

# Base de datos - SQL Server (store procedures).

- `mssql_init`.- Asocia a un store procedure.

Ejemplo: `$stmt=mssql_init("procedure", $con)`.

- `mssql_bind`.- Ingresa un parámetro del store procedure.

Ejemplo 1:

`mssql_bind($stmt, "@par1", ("&" si es output) $var, tipos (SQLVARCHAR, SQLINT4, SQLFLT8, etc), parámetro_output (true-false))`.

Ejemplo 2:

`mssql_bind($stmt, "@par1", &$var, SQLFLT8, TRUE)`.

# Base de datos - SQL Server (store procedures).

\*.- Para conseguir el return del store procedure se usa el siguiente parametro:

`mssql_bind($stmt, "RETVAL", &$var, tipo).`

- `mssql_execute.`- Ejecuta el store procedure.

Ejemplo: `$result=mssql_execute($stmt).`

# Manejo de cookies y sesiones.

Funciones relacionadas a los cookies:

- `setcookie`.- Crea un cookie en el cliente.

Ejemplo 1: `setcookie($nombre, $valor, $fecha)`.

Para eliminar un cookie basta con colocarle una fecha negativa o una cadena vacía como valor.

Ejemplo 2: `setcookie($nombre, "", time()-1)`.

- `$HTTP_COOKIE_VARS`.- Devuelve el valor de un cookie.

Ejemplo: `$HTTP_COOKIE_VARS[$valor]`.

# Manejo de cookies y sesiones.

Funciones relacionadas a las sesiones:

- `session_start`.- Comienza una sesión.
- `session_unset()`.- Elimina todas las variables de la sesión.
- `session_destroy`.- Elimina la sesión y del archivo temporal donde la almacena, siempre se debe invocar primero a `session_unset()`.
- `session_register`.- Registra una variable en una sesión.

Ejemplo: `session_register("var")`.

- `session_unregister`.- Elimina una variable de sesión.

Ejemplo: `session_unregister("var")`.

- `$HTTP_SESSION_VARS`.- Devuelve una variable de una sesión.

Ejemplo: `$HTTP_SESSION_VARS["var"] = valor`.

- `session_is_registered`.- Devuelve true si una variable existe en la sesión.

Ejemplo: `session_is_registered("var")`.

\*. En todos los casos siempre debe llamarse a la función `session_start()`.

# Manejo de cookies y sesiones (configuración).

PHP utiliza dos métodos para identificar la sesión, mediante un cookie con un identificador único de la sesión, o bien mediante un parámetro: SID, el cual contiene éste identificador. Si el navegador no dispone de cookies éste último será el método utilizado por PHP; si está activado: ([Session]>session.use\_trans\_sid).

Por esto debemos habilitar las siguientes opciones en el archivo PHP.INI:

```
session.use_cookies = 1  
session.use_trans_sid = 1.
```

# Web Services - Conceptos.

- XML-RPC.- Es un protocolo de llamada a procedimientos remotos, el cual trabaja sobre Internet.
- SOAP.- Es un protocolo basado en XML e indica como se deben codificar los mensajes que circulan entre 2 aplicaciones.
  - \*. Observaciones.- XML-RPC es sencillo y SOAP está diseñado para ofrecer un soporte completo de todo tipo de servicios web. Ambas no trabajan juntas.
- WSDL.- Es un formato XML que permite describir las funciones (con sus parámetros) que estarán disponibles.
- UDDI.- Permite ubicar servicios web en Internet.

# Web Services - Cliente.

## Pasos:

1. `include("lib/nusoap.php").`- Carga la librería nusoap.
  2. `$cliente = new soapclient("ruta web?wsdl", "wsdl").`- Consigue un objeto soap cliente.
  3. `$proxy = $cliente->getProxy().`- Consigue una clase proxy para llamar a las funciones.
  4. `$resultado = $proxy->función(par1...parn).`- Llama a la función expuesta.
  5. `$cliente->getError().`- Consigue el error obtenido.
- \*. Para trabajar con Web Services en PHP se utiliza la siguiente librería gratuita de la empresa NuSphere.

[Download](#)

# Web Services - Servidor.

## Pasos:

1. `include("lib/nusoap.php")`.- Carga la librería nusoap.
2. `$servidor = new soap_server()`.- Crea un objeto soap server.

3. Generación del WSDL.

```
$servidor->debug_flag = false;
```

```
$servidor->configureWSDL("Nombre del servicio",  
"http://".$_SERVER["SERVER_NAME"].$_SERVER["SCRIPT_NAME"]."\ruta web");
```

```
$servidor->wsdl->schemaTargetNamespace =  
"http://".$_SERVER["SERVER_NAME"].$_SERVER["SCRIPT_NAME"]."\ruta web").
```

# Web Services - Servidor (estructuras complejas)

## 4. Creación de registros:

```
$servidor->wsdl->addComplexType("TipoNuevo", "complexType",  
"struct", "all", "",  
array(  
"campo1" => array("name"=>"campo1", "type"=>"xsd:tipo_dato"),  
"campon" => array("name"=>"campon", "type"=>"xsd:tipo_dato"), )  
).
```

\*. Los principales tipos de datos son: xsd:string, xsd:decimal, xsd:integer, xsd:boolean, xsd:date, xsd:time, etc.

\*. También puede utilizarse un nuevo parámetro en los campos llamado default = "valor".

# Web Services - Servidor (estructuras complejas)

## 5. Creación de listas:

```
$servidor->wsdl->addComplexType("TipoNuevos", "complexType",  
"array", "", "SOAP-ENC:Array", array(),  
array(  
    array("ref" => "SOAP-ENC:arrayType",  
        "wsdl:arrayType" => "tns:TipoNuevo[]")  
    ),  
"tns:TipoNuevo"  
);
```

\*. Si se quiere referenciar tipos de datos complejos se usa "tns" y si se quiere referenciar tipos de datos simples se usa "xsd".

# Web Services - Servidor (registro de funciones)

```
6. $servidor->register("nombre_funcion",  
    array("par1"=>"tns:TipoNuevo",  
        "parn"=>"xsd:TipoSimple"),  
    array("return"=>"tns:TipoNuevos"),  
    "http://".$_SERVER["SERVER_NAME"].$_SERVER["S  
    CRIPT_NAME"]."\ruta web").
```

```
function nombre_funcion($par1, ..., $parn)  
{ ...  
    return ...; }
```

# Web Services - Servidor (enviar resultado).

7. Enviar el resultado como una respuesta SOAP por HTTP.

```
$HTTP_RAW_POST_DATA =  
isset($HTTP_RAW_POST_DATA) ?  
$HTTP_RAW_POST_DATA : "";  
$servidor->service($HTTP_RAW_POST_DATA);  
exit();
```

# Temas variados - COM.

Para instanciar objetos COM.

```
$obj = new COM("Clase COM").
```

Habilitar `com.allow_dcom` para instanciar objetos remotos.

# Temas variados - Java.

Para instanciar objetos Java.

```
$obj = new Java("Clase Java").
```

`java_last_exception_get()`.- Devuelve la excepción devuelta.

`java_last_exception_clear()`.- Limpia la última excepción.

# Temas variados - PDF.

Se usara la clase FPDF. [Download](#)

- El constructor acepta 3 parámetros:
  - 1er parámetro.- “P” (normal) y “L” (apaisado).
  - 2do parámetro.- “pt”, “mm”, “cm” y “in”.
  - 3er parámetro.- “A3”, “A4”, “A5”, “Letter” y “Legal”.
  - \*.- Por defecto se asume “P”, “mm” y “A4”.
- AddPage().- Agrega una página.

# Temas variados - PDF.

- `SetMargins(left, top, right)`.- Alinea los márgenes.
  - \*.- También se puede utilizar `SetLeftMargin(#)`, `SetRightMargin(#)` y `SetTopMargin(#)`.
  - \*.- Si no se define, el defecto es 1 en los márgenes.
- `SetFont`.- Se define el formato de letra, es obligatorio.
  - 1er parámetro.- Tipo de letra. (“Times”, “Courier”, “Symbol” y “Zapfdingbats”)
  - 2do parámetro.- “B” (negrita), “I” (cursiva) y “U” (subrayado).
  - 3er parámetro.- Tamaño de letra.

# Temas variados - PDF.

- Ln().- Realiza un salto de línea, recibe también el número de saltos por parámetro.
- GetStringWidth(\$texto).- Determina la longitud de una cadena en el tipo de letra actual.
- SetX(#).- Se posiciona en el eje X. Acepta valores negativos.
- SetY(#).- Se posiciona en el eje Y. Acepta valores negativos.
- PageNo().- Número de página.

# Temas variados - PDF.

- {nb}.- Devuelve el total de páginas, antes se debe usar el método AliasNbPage.
- Image.- Coloca una imagen en el archivo.
  - 1er parámetro.- Ruta del archivo.
  - 2do parámetro.- Posición en X.
  - 3er parámetro.- Posición en Y.
  - 4to parámetro.- Ancho de la imagen.
  - 5to parámetro.- Altura de la imagen.
- Write.- Escribe en el documento.
  - 1er parámetro.- Altura del texto.

# Temas variados - PDF.

2do parámetro.- El texto a mostrar.

3er parámetro.- Objeto hipervínculo (opcional).

- AddLink().- Consigue un objeto link.
- SetLink(\$objLink).- Envía al objeto link enviado por parámetro a la página actual.
- Cell.- Imprime una celda.

1er parámetro.- Posición en X.

2do parámetro.- Posición en Y.

3er parámetro.- El texto a mostrar.

4to parámetro.- 0 si no se desea mostrar un borde o 1 si se desea mostrar.

# Temas variados - PDF.

5to parámetro.- Número de saltos.

6to parámetro.- “R” (derecha), “C” (centro) y “L” (izquierda).

- SetDrawColor(r, g, b).- Define el color a pintar.
- SetFillColor(r, g, b).- Define el color de llenado.
- SetTextColor(r, g, b).- Define el color de los textos.
- SetLineWidth(#).- Define el ancho de las líneas.
- Output.- Envía el archivo al cliente.

1er parámetro.- El nombre del archivo.

# Temas variados - PDF.

2do parámetro.- “F” (Descarga el archivo automáticamente), “D” (Pide una ruta al cliente) e “I” (Pide una ruta al cliente, pero le agrega la extensión).

\*.- Si no enviamos valores, se mostrará el documento pdf en la estación cliente y no se guardará.

- Header() - Footer().- Se deben sobre escribir estos 2 métodos si deseamos modificar las cabeceras o los pie de páginas.
- SetTitle(\$titulo).- Se ingresa el título del documento.
- SetAuthor(\$autor).- Se ingresa el autor del documento.

# Temas variados - Gráficos.

Librería: php\_gd2.dll.

Content-type: header("Content-type: image/gif").

Crear un objeto image:

1.- Enlazado a un archivo de extensión gif.

```
$img = imagecreatefromgif("archivo.gif").
```

2.- Un imagen en blanco con X de ancho e Y de alto.

```
$img = imagecreate(X, Y).
```

Conseguir sus propiedades:

- `imagesx($img)`.- Consigue el ancho de la imagen.
- `imagesy($img)`.- Consigue el alto de la imagen.

# Temas variados - Gráficos.

- `imagecolorallocate($img, R, G, B)`.- Define un color, además dicha función devuelve un entero asociado al color.
- `imageline($img, X1, Y1, X2, Y2, $color)`.- Pinta una línea.
- `imagedashedline($img, X1, Y1, X2, Y2, $color)`.- Pinta una línea discontinua.
- `imagerectangle($img, X1, Y1, X2, Y2, $color)`.- Pinta un rectángulo.
- `imagearc($img, centroX, centroY, ancho, alto, ángulo inicio, ángulo final, $color)`.- Pinta un arco.
- `imagefill($img, X, Y, $color)`.- Pinta una dibujo geométrico por dentro pero no su borde.
- `imagefilltoborder($img, X, Y, $colorBorde, $colorFondo)`.- Pinta una dibujo geométrico por dentro y su borde.

# Temas variados - Gráficos.

- `imagestring($img, $tamaño, X, Y, $texto, $color)`.- Imprime un texto horizontalmente. Su tamaño varia de 1 a 5.
- `imagestringup($img, $tamaño, X, Y, $texto, $color)`.- Imprime un texto verticalmente. Su tamaño varia de 1 a 5.
- `imagecopyresized($imgd, $imgo, Xd, Yd, Xo, Yo, Anchod, Altod, Anchoo, Altoo)`.- Copia un conjunto de píxeles de un gráfico a otro. Donde d = destino, o = origen.
- `imagedestroy($img)`.- Libera el recurso.
- Envío al cliente:
  - 1.- Si se desea mostrar en el navegador la imagen:  
`imagegif($img)`.
  - 2.- Si se desea grabar en un directorio:  
`imagegif($img, "ruta")`.

# Temas variados - Expresiones regulares.

- $^X$  = Debe comenzar el texto con X. Si es acompañado de un  $[]$  indica una negación. Ejemplo:  $[^X]$ .
- $X\$$  = El texto debe finalizar en X.
- $[]$  = Permite ingresar una lista de letras.  
Ejemplo 1:  $[a-z]$ .  
Ejemplo 2:  $[a,b,c]$ .
- $.$  = Indica cualquier letra.  
Ejemplo: “ $X.X$ ” = Indica que entre ambos X puede existir cualquier letra.
- $X_+$  = Al menos 1 vez debe estar X en el texto.
- $X?$  = Puede existir en el texto 1 o 0 veces X.

# Temas variados - Expresiones regulares.

- $(X)\{n\}$  = N veces debe estar X en el texto.
- $\backslash$  = Caracter especial, convierte en caracter un comando.  
Ejemplo:  $\backslash^$ , esto reemplaza el comando  $^$  por su caracter.
- $()$  = Todo lo que este incluido dentro de el paréntesis se convierte en un sub patrón y devuelve un valor.
- Comandos especiales:  $[:blank:]$ ,  $[:digit:]$  y  $[:space:]$ .
- $\text{ereg}(\$patron, \$texto, \$coincidencias)$ .- Evalúa el texto según el patrón, es true si el texto cumple con el patrón.  
Si existe coincidencias los devuelve en un arreglo.

# Temas variados - Manejo de directorios.

- `chdir($ruta)`.- Posiciona en un directorio.
- `mkdir($ruta, $permisos)`.- Crea un directorio.
- `rmdir($ruta)`.- Elimina un directorio.
- `copy($rutao, $rutad)`.- Copia un archivo. O = origen, d = destino.
- `unlink($archivo)`.- Elimina un archivo.
- `rename(archivov, archivon)`.- Renombra un archivo. V = viejo, n = nuevo.
- `file_exists($ruta)`.- Devuelve true si existe el archivo.
- `file_size($ruta)`.- Devuelve el tamaño de un archivo.

# Temas variados - Manejo de directorios.

- `is_dir($ruta)`.- Devuelve true si la ruta es un directorio.
- `is_file($ruta)`.- Devuelve true si la ruta es un archivo.
- `is_readable("ruta")`.- Devuelve true si se puede leer el archivo o directorio.
- `is_writable("ruta")`.- Devuelve true si se puede escribir el archivo o directorio.
- `is_executable("ruta")`.- Devuelve true si se puede ejecutar el archivo.

# Temas variados - Manejo de directorios (clase DIR).

- Instanciar una clase DIR: `$obj = dir("ruta")`.
- Path.- Devuelve la ubicación donde apunta el objeto.  
Ejemplo: `$obj->path`.
- Read.- Devuelve un elemento de la ubicación donde apunta el objeto, en caso de no existir más elementos, la función retorna false.

Ejemplo: `while ($elemento = $obj->read()) { ... }`

FIN