

## VeryTinyAJAX2 0.2

Wrapper JavaScript simple a funciones XMLHttpRequest para AJAX e integración con PHP en busca de la reducción y simplificación de código.

Licencia: GPLv2 (<http://www.gnu.org/licenses/gpl.txt>)

Autores.: Pablo Rodríguez Rey (mr.xkr -en- inertinc -punto- org)

<http://mr.xkr.inertinc.org/>

Javier Gil Motos (cucaracha -en- inertinc -punto- org)

<http://cucaracha.inertinc.org/>

### Funciones JavaScript integradas en *ajax.js*:

Función	Ejemplo de uso	Salida
ajaxVersion()	alert(ajaxVersion());	VeryTinyAJAX2/0.2
gid(id)	alert(gid("identificador"));	[Object] si existe document.getElementById("identificador"), o el mismo object si ya lo era previamente.
httpObject()	alert(httpObject());	[Object] si soporta AJAX, o false en caso contrario.
httpStateString(readyState)	alert(httpStateString(4));	Complete, texto de estado que indique el código de estado
ajaxEnabled()	alert(ajaxEnabled());	true/false dependiendo de si el navegador soporta AJAX o no
gescape(cadena)	alert(gescape("me&you"));	me%26you - pasa a formato url-encoded caracteres especiales
hescape(cadena)	alert(gescape("me³you"));	me%b3you - pasa a formato url-encoded caracteres <32 o >127.
jescape(cadena)	<b>Uso interno</b>	Convierte caracteres especiales para ser usados para codificar variables en formato JSON.
json(variable)	json({"hola":"ejemplo"});	{"hola":"ejemplo"} en formato cadena
adump(variable)	adump({"hola":"ejemplo","valor":12});	"hola" = string("ejemplo") "valor" = number("12") devuelve una cadena que representa la variable y sus tipos para ser leída
ajax(a,[b],[c],[d]);	<b>Ver abajo</b>	Realiza una petición AJAX al servidor. Ver documentación más abajo.
xf.value(inputobject)	alert(xf.value("objeto"));	VALOR - helper que devuelve el valor de un campo tipo INPUT, TEXTAREA, SELECT, CHECKBOX y OPTION.
xf.enabled(formobject)	xf.enabled(false);	Habilita o deshabilita un formulario para que no puedan modificarlo o reenviarlo.
xf.form(formobject)	alert(xf.form("formulario"));	["campo1":"valor1","campo2":"valor2"] - devuelve todos los campos de un formulario en formato JSON.
xf.fields(campos)	alert(xf.fields("campo1,campo2")); alert(xf.fields(["campo1","campo2"])); <b>(ambas son equivalentes)</b>	["campo1":"valor1","campo2":"valor2"] - devuelve los campos seleccionados por su identificador en formato JSON.

## Funciones PHP integradas en *ajax.php*:

Función	Ejemplo de uso	Salida
<code>ajson(variable)</code>	<code>echo ajson(Array(1,2,3,4));</code>	<code>[1,2,3,4]</code> - Convierte a JSON una variable PHP. Válido para charset que no sean UTF-8 (elimina la limitación de <code>json_encode</code> ) - evitar su uso.
<code>aget([datos])</code>	<code>aget(\$_POST["data"]);</code>	- Decodifica los valores previamente codificados con JSON. - Si se llama sin parámetros, equivale a obtener <code>\$adata</code> , por tanto, <code>\$adata=aget();</code>
<code>aput(nombre,valor);</code>	<code>aput("nombre","valor");</code>	Acumula un valor para luego enviar de forma transaccional con <code>ajax()</code> .
<code>ajax([datos]);</code>	<b>Ver abajo</b>	Realiza una petición AJAX al servidor. Ver documentación más abajo.

## Parámetros y formas de llamar a la función `ajax()` en **JavaScript**:

Envío JavaScript	Devuelve
<b>Enviar datos por GET usando formato estándar:</b> <pre>ajax({   "get":{     "dato1":valor1     +"&amp;dato2="+gescape("valor2")     +"&amp;daton="+gescape("valorn")   } });</pre>	<b>En el servidor (PHP):</b> <pre>\$_GET["dato1"]="valor1"; \$_GET["dato2"]="valor2"; \$_GET["daton"]="valorn";</pre>
<b>Enviar datos por POST usando formato estándar:</b> <pre>ajax({   "post":{     "dato1":valor1     +"&amp;dato2="+gescape("valor2")     +"&amp;daton="+gescape("valorn")   } });</pre>	<b>En el servidor (PHP):</b> <pre>\$_POST["dato1"]="valor1"; \$_POST["dato2"]="valor2"; \$_POST["daton"]="valorn";</pre>
<b>Enviar datos por GET usando JSON:</b> <pre>ajax({   "get":{     "dato1":"valor1",     "dato2":"valor2",     "daton":"valorn"   } });</pre>	<b>En el servidor (PHP):</b> <pre>\$_GET["dato1"]="valor1"; \$_GET["dato2"]="valor2"; \$_GET["daton"]="valorn";</pre>
<b>Enviar datos por POST usando JSON:</b> <pre>ajax({   "post":{     "dato1":"valor1",     "dato2":"valor2",     "daton":"valorn"   } });</pre>	<b>En el servidor (PHP):</b> <pre>\$_POST["dato1"]="valor1"; \$_POST["dato2"]="valor2"; \$_POST["daton"]="valorn";</pre>
<b>Cambiar la URL y añadir datos GET:</b> <pre>ajax({   "url":"?dato1=valor1&amp;dato2=valor2"   +"&amp;daton="+gescape("valorn") });</pre>	<b>En el servidor (PHP):</b> <pre>\$_GET["dato1"]="valor1"; \$_GET["dato2"]="valor2"; \$_GET["daton"]="valorn";</pre>
<b>Recepción de datos asíncrona:</b> <pre>ajax({   "get":{"dato1=valor1",   "async":function(r){     alert(adump(r));   } });</pre> <b>Recepción de datos síncrona:</b> <pre>ajax({   "get":{"dato1=valor1",   "sync":function(r){     alert(adump(r));   } });</pre>	<b>En el servidor (PHP):</b> <pre>\$_GET["dato1"]="valor1";</pre> <b>En el cliente se recibe:</b> <pre>"state" = number("4") "stateString" = string("Complete") "complete" = boolean("true") "status" = number("200") "text" = string("DATOS DEL SERVIDOR") "xml" = object(0): "data" = object(0): "error" = boolean("true")</pre>
<b>Petición abreviada de GET con ajax=valor:</b> <pre>ajax({   "ajax":"valor", });</pre>	<b>En el servidor (PHP):</b> <pre>\$_GET["ajax"]="valor";</pre> <b>Y de forma automatizada:</b> <pre>\$ajax="valor";</pre>

Envío JavaScript	Devuelve
<p><b>Petición abreviada de POST con datos:</b></p> <pre>ajax({   "data":{     "dato1":"valor1",     "dato2":"valor2",     "daton":"valorn"   } });</pre>	<p><b>En el servidor (PHP):</b>  \$_POST["data"]={'dato1':"valor1","dato2':"valor2","daton':"valorn"};</p> <p><b>De forma automatizada:</b>  \$data=Array(   "dato1"=&gt;"valor1",   "dato2"=&gt;"valor2",   "daton"=&gt;"valorn" )</p>
<p><b>Mostrar mensaje de error al encontrarse uno:</b></p> <pre>ajax({   "url":"inexistente",   "error":true });</pre>	<pre>newalert("Se ha encontrado el error 404 en el servidor") si existe esta función, o alert("Se ha encontrado el error 404 en el servidor")</pre>
<p><b>Mostrar mensaje de error al encontrarse uno:</b></p> <pre>ajax({   "url":"inexistente",   "error":"Mensaje de error" });</pre>	<pre>newalert("Mensaje de error") si existe esta función, o alert("Mensaje de error")</pre>
<p><b>Mostrar mensaje de error al encontrarse uno:</b></p> <pre>ajax({   "url":"inexistente",   "error":function(e){     alert(adump(e));   } });</pre>	<p><b>En el cliente se ejecuta el evento y se recibe:</b></p> <pre>"status" = number("404") "error" = string("Se ha encontrado el error 404 en el servidor.") "show" = function("function () {   if (typeof newalert == "function") {     newalert({ico: "error", id: "verytinyajax2", msg: this.error});   } else {     alert(this.error);   } }");</pre>
<p><b>Petición abreviada:</b></p> <pre>ajax("accion",{   "dato1":"valor1",   "dato2":"valor2",   "daton":"valorn" },function(r){   alert(adump(r)); });</pre> <p><b>Equivale a:</b></p> <pre>ajax({   "ajax":"accion",   "data":{     "dato1":"valor1",     "dato2":"valor2",     "daton":"valorn"   },   "async":function(r){     alert(adump(r));   } });</pre>	<p><b>En el servidor (PHP):</b>  \$_GET["ajax"]="accion";  \$_POST["data"]={'dato1':"valor1","dato2':"valor2","daton':"valorn"};</p> <p><b>Se obtiene de forma automatizada:</b>  \$ajax="accion";  \$data=Array(   "dato1"=&gt;"valor1",   "dato2"=&gt;"valor2",   "daton"=&gt;"valorn" );</p>
<p><b>Envío de un formulario completo abreviado:</b></p> <pre>ajax("accion",xf.form("id_formulario"),function(r) {   alert(adump(r)); });</pre>	<p><b>En el servidor (PHP):</b>  \$_GET["ajax"]="accion";  \$_POST["data"]={'dato1':"valor1","daton":"'valorn"};</p> <p><b>Se obtiene de forma automatizada:</b>  \$ajax="accion";  \$data=Array(   "dato1"=&gt;"valor1",   "daton"=&gt;"valorn" ); // suponiendo que estos sean los valores del formulario</p>



## Forma de llamar a la función `ajax()` en **PHP**:

Envío Servidor PHP:	En el cliente se recibe:
<b>Envío de datos AJAX desde PHP:</b> <code>ajax("cadena de datos");</code>	<b>En el cliente (JavaScript):</b> <code>r="cadena de datos";</code>
<b>Envío de datos AJAX desde PHP:</b> <code>ajax(Array("ok"=&gt;true));</code>	<b>En el cliente (JavaScript):</b> <code>r.ok=true;</code>
<b>Envío de datos AJAX desde PHP:</b> <code>ajax(Array("lista"=&gt;Array(1,2,3,4,5)));</code>	<b>En el cliente (JavaScript):</b> <code>r.lista=[1,2,3,4,5];</code>
<b>Envío de datos AJAX desde PHP:</b> <code>aput("ok",true);</code> <code>aput("lista",Array(1,2,3,4,5));</code> <code>ajax();</code>	<b>En el cliente (JavaScript):</b> <code>r.ok=true;</code> <code>r.lista=[1,2,3,4,5];</code>

Para ejemplos de uso, ver [ajax.test.php](#).

Adicionalmente, se ofrece el valor de `$me=$_SERVER["PHP_SELF"]`;, es decir, el mismo nombre del script que se está ejecutando en PHP en `$me`, útil para hacer referencias a si mismo en URLs.