Sessiones y Cookies

Session

Introducción
El soporte para sesiones en PHP consiste en una forma de preservar cierta información a lo largo de accesos subsiguientes.

A un visitante que accede a un sitio web se le asigna un id único, también llamado id de sesión.

session_start
(PHP 4, PHP 5, PHP 7)

session_start — Iniciar una nueva sesión o reanudar la existente

Descripción

session_start ([array $options = []]) : bool

session_start() crea una sesión o reanuda la actual basada en un identificador de sesión pasado mediante una petición GET o POST, o pasado mediante una cookie.

Cuando session_start() es llamada o cuando se autoinicia una sesión, PHP llamará a los gestores de almacenamiento de sesiones open y read. Éstos serán un gestor de almacenamiento proporcionado por omisión o por extensiones de PHP (como SQLite o Memcached); o pueden ser un gestor personalizado como está definido en session_set_save_handler(). La llamada de retorno read recuperará cualquier información se de sesión existente (almacenada en un formato serializado especial) y será deserializada y usada para rellenar automáticamente la variable superglobal $_SESSION cuando la llamada de retorno read devuelva la información de sesión guardada a la gestión de sesiones de PHP.

Para usar una sesión nominada llame a session_name() antes de llamar a session_start().

Cuando session.use_trans_sid está habilidado, la función session_start() registrará un gestor de salida interno para la reescritura de URL.

Si un usuario utiliza ob_gzhandler o similar con ob_start(), el orden de las funciones es importante para la salida adecuada. Por ejemplo, ob_gzhandler se debe registrar antes de iniciar una sesión.

Parámetros

options
Si se proporciona, este array asociativo de opciones sobrescribirá las directivas de configuración de sesiones establecidas actualmente. Las claves no deben incluir el prefijo session..

Además del conjunto habitual de directivas de configuración, también se puede proporcionar la opción read_and_close. Si se establece a TRUE, resultará en el cierre inmediato de la sesión después de haber sido leída, evitando así el bloqueo inncesario si los datos de la sesión no han cambiado.

Valores devueltos
Esta función devuleve TRUE si una sesión fue iniciada satisfactoriamente, si no, devuelve FALSE.

Ejemplos

Un ejemplo de sesión básico

Ejemplo #1 pagina1.php

```
<?php
// pagina1.php

session_start();

echo 'Bienvenido a la página #1';

$_SESSION['color']  = 'verde';
$_SESSION['animal'] = 'gato';
$_SESSION['instante']   = time();

// Funciona si la cookie de sesión fue aceptada
echo '<br /><a href="pagina2.php">página 2</a>';

// O quizás pasar el id de sesión, si fuera necesario
echo '<br /><a href="pagina2.php?' . SID . '">página 2</a>';
?>
```

$_SESSION
(PHP 4 >= 4.1.0, PHP 5, PHP 7)

$_SESSION -- Variables de sesión

Descripción
Es un array asociativo que contiene variables de sesión disponibles para el script actual. Ver la documentación de Funciones de sesión para más información sobre su uso.

Nota:
Esta es una 'superglobal' o una variable automatic global. Significa simplemente que es una variable que está disponible en cualquier parte del script. No hace falta hacer global $variable; para acceder a la misma desde funciones o métodos.

Ejemplo
```
<?php
session_start();
/*session is started if you don't write this line can't use $_Session
global variable*/
$_SESSION["newsession"]=$value;
```

```
?>
```

session_id
(PHP 4, PHP 5, PHP 7)

session_id — Obtener y/o establecer el id de sesión actual

Descripción

```
session_id ([ string $id ] ) : string
```

session_id() se usa para obtener o establecer el id de sesión para la sesión actual.

La constante SID también se puede usar para recuperar el nombre y la sesión actuales como una cadena apropiada para añadir a las URL. Véase también Manejo de Sesiones.

Parámetros

id
Si se especifica id, reemplazará el id de sesión actual. session_id() necesita ser llamado antes de session_start() para este propósito. Dependiendo del gestor de sesión, no todos los caracteres están permitidos dentro del id de sesión. Por ejemplo, el gestor de archivo de sesión ¡sólo permite caracteres en el rango a-z A-Z 0-9 , (coma) y - (menos)!

Nota:
Cuando se usan cookies de sesión, especificar un id para session_id() enviará siempre una nueva cookie cuando se llame a session_start(), sin importar si el id de sesión actual es idéntico al que se va a establecer.

Valores devueltos
session_id() devuelve el id de sesión para la sesión actual o la cadena vacía ("") si no hay sesión actual (no existe id de sesión actual).

session_name
(PHP 4, PHP 5, PHP 7)

session_name — Obtener y/o establecer el nombre de la sesión actual

Descripción

```
session_name ([ string $name ] ) : string
```

session_name() devuelve el nombre de la sesión actual. Si se da el nombre name, session_name() actualizará el nombre de la sesión y devolverá el nombre antiguo de la sesión.

El nombre de la sesión se reinicia al valor predeterminado almacenado en session.name en el momento de iniciar una petición. Por lo tanto,

se necesita llamar a session_name() por cada petición (y antes de llamar a session_start() o session_register()).

Parámetros

name
El nombre de la sesión hace referencia al nombre de la sesión usado cookies y URLs (p.ej. PHPSESSID). Debería contener sólo caracteres alfanuméricos; debería ser corto y descriptivo (esto es, para usuarios con las advertencias de cookies habilitadas). Si se especifica name, el nombre de la sesión actual se cambia por su valor.

Advertencia
El nombre de la sesión no puede consistir en dígitos solamente, debe de estar presente al menos una letra. De otro modo se genera un nuevo id de sesión cada vez.

Valores devueltos
Devuelve el nombre de la sesión actual. Si se proporciona name y una función actualiza el nombre de la sesión, se devolverá el nombre antiguo de la sesión.

Ejemplos

Ejemplo #1 Ejemplo de session_name()

```
<?php

/* establecer el nombre de la sesión a WebsiteID */

$nombre_anterior = session_name("WebsiteID");

echo "El nombre anterior de la sesión era $nombre_anterior<br />";
?>
```

session_encode
(PHP 4, PHP 5, PHP 7)

session_encode — Codifica los datos de la sesión actual como un string codificado de sesión

Descripción

session_encode (void) : string

session_encode() devuelve un string serializado del contenido de los datos de la sesión actual almacenados en el array superglobal $_SESSION.

Por defecto, el método de serialización usado es interno a PHP, y no es el mismo que serialize(). El método de serialización se puede establecer con session.serialize_handler.

Valores devueltos

Devuelve el contenido de la sesión actual codificado.

Notas

Advertencia
Se debe llamar a session_start() antes de usar session_encode().

session_decode
(PHP 4, PHP 5, PHP 7)

session_decode — Decodifica la información de sesión desde una cadena de sesión codificada

Descripción

session_decode (string $data) : bool

session_decode() decodifica la información de sesión serializada proporcianda en $data, y rellena la variable superglobal $_SESSION con el resultado.

Por defecto, el método de deserialización usado es interno a PHP, y no es el mismo que unserialize(). El método de serialización se puede establecer con session.serialize_handler.

Parámetros

data
Los datos codificados a almacenar.

Valores devueltos
Devuelve TRUE en caso de éxito o FALSE en caso de error.

session_destroy
(PHP 4, PHP 5, PHP 7)

session_destroy — Destruye toda la información registrada de una sesión

Descripción

session_destroy (void) : bool

session_destroy() destruye toda la información asociada con la sesión actual. No destruye ninguna de las variables globales asociadas con la sesión, ni destruye la cookie de sesión. Para volver a utilizar las variables de sesión se debe llamar a session_start().

Para destruir la sesión completamente, como desconectar al usuario, el id de sesión también debe ser destruido. Si se usa una cookie para propagar el id de sesión (comportamiento por defecto), entonces la cookie de sesión se debe borrar. setcookie() se puede usar para eso.

Valores devueltos
Devuelve TRUE en caso de éxito o FALSE en caso de error.

Ejemplos

Ejemplo #1 Destruir una sesión con $_SESSION

```
<?php
// Inicializar la sesión.
// Si está usando session_name("algo"), ¡no lo olvide ahora!
session_start();

// Destruir todas las variables de sesión.
$_SESSION = array();

// Si se desea destruir la sesión completamente, borre también la
cookie de sesión.
// Nota: ¡Esto destruirá la sesión, y no la información de la sesión!
if (ini_get("session.use_cookies")) {
    $params = session_get_cookie_params();
    setcookie(session_name(), '', time() - 42000,
        $params["path"], $params["domain"],
        $params["secure"], $params["httponly"]
    );
}

// Finalmente, destruir la sesión.
session_destroy();
?>
```

Cookies

$_COOKIE
$HTTP_COOKIE_VARS [obsoleta]

(PHP 4 >= 4.1.0, PHP 5, PHP 7)

```
$_COOKIE[‘nombre’];
```

Descripción

Una variable tipo array asociativo de variables pasadas al script actual a través de Cookies HTTP.

Ejemplos

Ejemplo #1 Ejemplo de $_COOKIE

```
<?php
echo '¡Hola ' . htmlspecialchars($_COOKIE["nombre"]) . '!';
?>
```

Asumiendo que la cookie "nombre" ha sido definida anteriormente

El resultado del ejemplo sería algo similar a:

¡Hola Juan!

setcookie

(PHP 4, PHP 5, PHP 7)

setcookie — Enviar una cookie

Descripción

setcookie (string $name [, string $value = "" [, int $expires = 0 [, string $path = "" [, string $domain = "" [, bool $secure = FALSE [, bool $httponly = FALSE]]]]]) : bool

setcookie (string $name [, string $value = "" [, array $options = []]]) : bool

setcookie() define una cookie para ser enviada junto con el resto de cabeceras HTTP. Como otros encabezados, cookies deben ser enviadas antes de cualquier salida en el script (este es un protocolo de restricción). Esto requiere que hagas llamadas a esta función antes de cualquier salida, incluyendo etiquetas <html> y <head> así como cualquier espacio en blanco.

Una vez que las cookies se han establecido, se puede acceder a ellas en la siguiente página de carga con el array $_COOKIE. valores Cookie también pueden existir en $_REQUEST.

Parámetros

» RFC 6265 proporciona la referencia normativa como cada parámetro setcookie() es interpretado.

name

El nombre de la cookie.

value

El valor de la cookie. Este valor se almacena en el ordenador del cliente; no almacenar información sensible. Asumiendo que el name es 'cookiename', este valor es recuperado a través de $_COOKIE['cookiename']
expires

El tiempo en que la cookie expira. Esta es una marca de tiempo Unix en número de segundos desde la época. En otras palabras, lo más

probable es que se haga con la función time() más el número de segundos antes de que quiera que expire. O se podría usar mktime(). time()+60*60*24*30 hará que la cookie establecida expire en 30 días. Si se establece a 0, o es omitido, la cookie expirará al final de la sesión (cuando el navegador es cerrado).

Nota:

Puede notar que el parámetro expires toma una marca de tiempo Unix, en lugar de formato de fecha Wdy, DD-Mon-YYYY HH:MM:SS GMT, esto es porque PHP hace la conversión internamente.

path

La ruta dentro del servidor en la que la cookie estará disponible. Si se utiliza '/', la cookie estará disponible en la totalidad del domain. Si se configura como '/foo/', la cookie sólo estará disponible dentro del directorio /foo/ y todos sus sub-directorios en el domain, tales como /foo/bar/. El valor por defecto es el directorio actual en donde se está configurando la cookie. domain

El (sub)dominio al que la cookie está disponible. Estableciendo esto a un subdominio (como 'www.example.com') hará que la cookie esté disponible para ese subdominio y todos los demás subdominios del mismo (p.e. w2.www.example.com). Para que la cookie esté disponible para todo el dominio (incluyendo todos sus subdominios), simplemente establezca el nombre de domonio ('example.com', en este caso).

Los navegadores más antiguos todavía implementan la obsoleta » RFC 2109 pueden necesitar un . para comparar todos los subdominios.

secure

Indica que la cookie sólo debiera transmitirse por una conexión segura HTTPS desde el cliente. Cuando se configura como TRUE, la cookie sólo se creará si es que existe una conexión segura. Del lado del servidor, depende del programador el enviar este tipo de cookies solamente a través de conexiones seguras (por ejemplo, con $_SERVER["HTTPS"]).

httponly

Cuando es TRUE la cookie será accesible sólo a través del protocolo HTTP. Esto significa que la cookie no será accesible por lenguajes de scripting, como JavaScript. Se ha indicado que esta configuración ayuda efectivamente a reducir el robo de identidad a través de ataques XSS (aunque no es soportada por todos los navegadores). pero esa afirmación se disputa a menudo. Agregado en PHP 5.2.0. Puede ser TRUE o FALSE

options

Un array asociativo que puede tener cualquiera de las claves expires, path, domain, secure, httponly y samesite. Los valores tienen el mismo significado que se describe para los parámetros con el mismo nombre. El valor de el elemento samesite debería ser None, Lax o Strict. Si no se da ninguna de las opciones permitidas, sus valores por defecto son los mismos que los valores por defecto de los parámetros explícitos. Si el elemento samesite es omitido, no se establece ningún atributo de la cookie SameSite.

Valores devueltos

Si existe algún tipo de output anterior a la llamada de esta función, setcookie() fallará y retornará FALSE. Si setcookie() ejecuta satisfactoriamente, retornará TRUE. Esto no indica si es que el usuario ha aceptado la cookie o no.

Ejemplos

A continuación se presentan algunos ejemplos de cómo enviar cookies:

Ejemplo #1 Ejemplo de envío con setcookie()

```
<?php
$value = 'something from somewhere';

setcookie("TestCookie", $value);
setcookie("TestCookie", $value, time()+3600);  /* expira en 1 hora */
setcookie("TestCookie", $value, time()+3600, "/~rasmus/",
"example.com", 1);
?>
```

Nótese que la parte del valor de la cookie será automáticamente codificada con urlencode al enviar la cookie, y al ser recibida será automáticamente decodificada y asignada a una variable con el mismo nombre que el nombre de la cookie. Si no se desea ésto, se puede usar setrawcookie() si es que se está utilizando PHP 5. Para ver el contenido de nuestra cookie de prueba en un script, simplemente siga uno de los ejemplo siguientes:

```
<?php
// Imprime una cookie individual
echo $_COOKIE["TestCookie"];

// Otra forma de depurar/prueba es ver todas las cookies
print_r($_COOKIE);
?>
```

Ejemplo #2 Ejemplo de eliminación con setcookie()

Al borrar una cookie debe asegurarse que la fecha de expiración ya ha pasado, de modo a detonar el mecanismo de eliminación del navegador. El siguiente ejemplo muestra cómo borrar las cookies enviadas en el ejemplo anterior:

```
<?php
// establecer la fecha de expiración a una hora atrás
setcookie("TestCookie", "", time() - 3600);
setcookie("TestCookie", "", time() - 3600, "/~rasmus/",
"example.com", 1);
?>
```

Ejemplo #3 setcookie() y los arrays

También puede crear arrays de cookies utilizando la notación de arrays en el nombre de la cookie. El efecto de esto es de crear tantas cookies como elementos hay en el array, pero al recibir el script la cookie, todos los valores son colocados en un array con el nombre de la cookie:

```
<?php
// set the cookies
setcookie("cookie[three]", "cookiethree");
setcookie("cookie[two]", "cookietwo");
setcookie("cookie[one]", "cookieone");

// después de que la página se recargue, imprime
if (isset($_COOKIE['cookie'])) {
    foreach ($_COOKIE['cookie'] as $name => $value) {
        $name = htmlspecialchars($name);
        $value = htmlspecialchars($value);
        echo "$name : $value <br />\n";
    }
}
?>
```

El resultado del ejemplo sería:

```
three : cookiethree
two : cookietwo
one : cookieone
```