Capítulo 19: Funciones adicionales de PHP

Persistencia:

Una de las características importantes en lenguajes orientados a objetos o lenguajes de scripting modernos es la persistencia, un objeto persistente es aquel que puede almacenarse en un medio de almacenamiento secundario (un archivo o una base de datos) para luego recuperarlo. PHP provee de dos funciones que permiten realizar esto serializando y des-serializando variables de PHP.

string=serialize(var);

Recibe cualquier variable de PHP incluso un objeto y devuelve un string que es una representación de la variable, dicho string puede almacenarse en un archivo o una base de datos para lograr persistencia.

var=unserialize(string);

Recibe un string que es la serialización de una variable, des-serializa y asigna a la variable pasada. Para des-serializar un objeto es necesario que el script que usa unserialize disponga de la definición de la clase.

Funciones de hashing y encriptación:

string=md5(string)

Devuelve un string de 32 bytes que es un "digest" del string original, es decir aplica al string original una función de hashing unidireccional.

string=crypt(string)

Encripta un string usando el método unidireccional de Unix, usado por ejemplo para almacenar passwords, el string devuelto es de extensión variable. No se puede desencriptar.

Generación de identificadores únicos:

string=uniqid(string_base);

Construye un identificador único tomando como base un string pasado, por ejemplo para generar identificadores únicos de 32 bits aleatorios se usa:

$better_token = md5 (uniqid (rand()));

Ejecución de codigo PHP:

eval(string_codigo);

Evalúa el string como si fuera código PHP y lo ejecuta.

Ejemplo:

eval("print(\"hola\")");

Imprime hola como si se ejecutara la instrucción print, la función eval es útil en varios casos por ejemplo para guardar código PHP en un archivo o base de datos y luego recuperarlo y ejecutarlo dinámicamente (por ejemplo para que usuarios de un web site suban sus propios scripts PHP) o bien usando funciones de parsing XML para insertar en XML processing-instructions de tipo <?php código ?> y luego en el script php que parsea el XML ejecutar el código php con eval.

Control del script:

sleep(segundos);

Hace una pause de la cantidad de segundos indicados.

die(string);

Termina la ejecución del script imprimiendo el string pasado.

exit();

Finaliza la ejecución del script.

Manejo de URLs

string=base64_decode(string);

Decodifica un string encodeado en base64.

string=base64_encode(string);

Codifica un string a base64.

string=urlencode(string);

Codifica un string de acuerdo a RFC1738 es decir reemplaza todos los caracteres que no sean alfanuméricos o "_","."," por un signo "%" seguido de una representación hexadecimal del caracter de acuerdo a RFC1738, los espacios en blanco se codifican como %20 por ejemplo. Este formato es "seguro" para pasarlo como query_string en un URL

Ejemplo tipico:

```
$ulr_string=urlencode($string_raro);
print("<a href=\"http://algo.com?$url_string\">");
etc...
```

string=urldecode(string);

Decodifica un string encodeado con urlencode.

array=parse_url(string_url);

Recibe un string representando una URL y devuelve un vector asociativo de donde pueden recuperarse las sigiuientes claves:

"scheme", "host", "port", "path", "query"

Ejemplo:
http://www.yahoo.com:8080/pruebas/coso.php?hola=5&cosa=6

Scheme : http
Host : www.yahoo.com
Port : 8080
Path : /pruebas/coso.php
Query : hola=5&cosa=6