Published on *Marco de Desarrollo de la Junta de Andalucía* (http://madeja.i-administracion.junta-andalucia.es/servicios/madeja)

Reglas de construcción en PHP

- **Área:** Especificaciones de Codificación y Construcción
- **Tipo de pauta:** Directriz
- **Carácter de la pauta:** Obligatoria

Código: LIBP-0102

A continuación se muestran una indicaciones que se deben seguir para estandarizar la construcción de software en PHP

Existen muchos estudios sobre la eficiencia en la ejecución de determinadas instrucciones en PHP. Es necesario estandarizar la construcción de software en PHP optimizando el tiempo de ejecución y favoreciendo la legibilidad del código. A continuación se van a ofrecer una serie de recomendaciones orientadas a mejorar la eficiencia, la legibilidad y la calidad del código construido en PHP.

Pautas

Función eval()

No utilizar la función **eval()**

Se prohibe el uso de la función **eval()** cuando sea posible y cuando no lo sea, asegurar que los datos usados en la construcción no hayan podido ser manipulados.

Volver al índice

Función phpinfo()

Desactivar la función **phpinfo()**

Se debe desactivar la función **phpinfo()**, por motivos de seguridad, y restringir el acceso a cualquier recurso que utiliza esta función, ya que expone el contenido de la matriz **$_SERVER**, lo que facilitaría información para proyectar ataques sobre la aplicación.

Volver al índice

Función die()

No usar **die()** para el manejo de errores

Si se produce un error, éste no reportará información concluyente, provocando un sensación poco agradable al usuario. Se deben utilizar funciones como **trigger_error()** junto con **set_error_handler()** para manejar los errores de una aplicación.

Volver al índice

Funciones de conteo en bucles

No usar funciones de conteo dentro de bucles

No se deben utilizar funciones tipo **count()** que se ejecuten dentro de la iteración, porque penalizan la velocidad del proceso. Si se calcula la función fuera del bucle, mejora la velocidad en torno a un 600% respecto a la codificación inicial.

Volver al índice

Métodos estáticos

Declarar los métodos estáticos cuando sea posible

Si es posible, hay que declarar los métodos como estáticos, siempre que vayan a ser tratados de esta manera. Esta probado que se reduce su tiempo de ejecución hasta cuatro veces con respecto a los métodos que no están declarados como estáticos.

Volver al índice

Función time()

Utilizar la variable **$_SERVER ['REQUEST_TIME']** sobre la función **time()**

Cuando necesite calcular el tiempo actual dentro de la ejecución del algún script en una aplicación es mucho más eficiente utilizar **$_SERVER ['REQUEST_TIME']** que la función **time()**.

Volver al índice

Función foreach()

Utilizar **foreach()** en los bucles de colecciones y arrays

Cuando tenemos una estructura con un bucle destinado a la lectura de un array o una colección está demostrado que es mucho más rápido utilizar **foreach()** antes que estructuras del tipo **while** o **for**.

Volver al índice

Función strpos()

Utilizar **strpos()** para las búsquedas de subcadenas

Para realizar la búsqueda de subcadenas dentro de cadenas de texto, la mejor manera de realizarlo es utilizar la función **strpos()**, preferentemente sobre **preg_match()** o **ereg()**.

Volver al índice

Preincremento de variables

Realizar preincremento de las variables cuando sea posible

Realizar un preincremento de una variable **++$i** es más rápido que el postincremento **$i++** de esa variable. Se deben cambiar las instrucciones para que se realice el preincremento cuando se pueda y sobre todo en bucles críticos.

Volver al índice

Depuración de código

Realizar depuraciones de código

Se debe revisar cuidadosamente la salida que las páginas PHP producen con el fin de asegurarse de que no se están produciendo errores. Como alternativa, puede asegurarse que el error de registro esté activado en su archivo php.ini, y comprobarlo con regularidad. Existen funciones que aumentan la capacidad de depuración, permitiendo comprobar los scripts sin necesidad de mostrar datos en HTML.

Volver al índice

Conexiones persistentes

Usar conexiones persistentes

Se debe conectar a la base de datos mediante conexiones persistentes. Éstas permanecen aunque el script haya terminado, lo que agiliza la ejecución de otro script si conecta a base de datos porque la conexión sigue abierta. Esto ahorra tiempo negociando contraseñas y la ejecución de parte del código.

Volver al índice

Inicialización de variables

Inicializar las variables para reducir el tiempo de ejecución

Aunque no es necesario, se recomienda inicializar las variables en PHP para reducir el tiempo de ejecución. Las variables no inicializadas tienen un valor predeterminado de acuerdo a su tipo (las booleanas se asumen como FALSE, los enteros y flotantes como cero, las cadenas se establecen como una cadena vacía y las matrices se convierten en un array vacío). El algunos casos, como los métodos GET, no será posible inicializar las variables.

Volver al índice

Recursos

Área: Desarrollo » Especificaciones de Codificación y Construcción

Código	Título	Tipo	Carácter
RECU-0764	Implementación de reglas de construcción en PHP	Ejemplo	Obligatorio
RECU-0256	Manual de PHP	Manual	Recomendado
RECU-0255	PHPDocumentor	Ficha Técnica	Recomendado