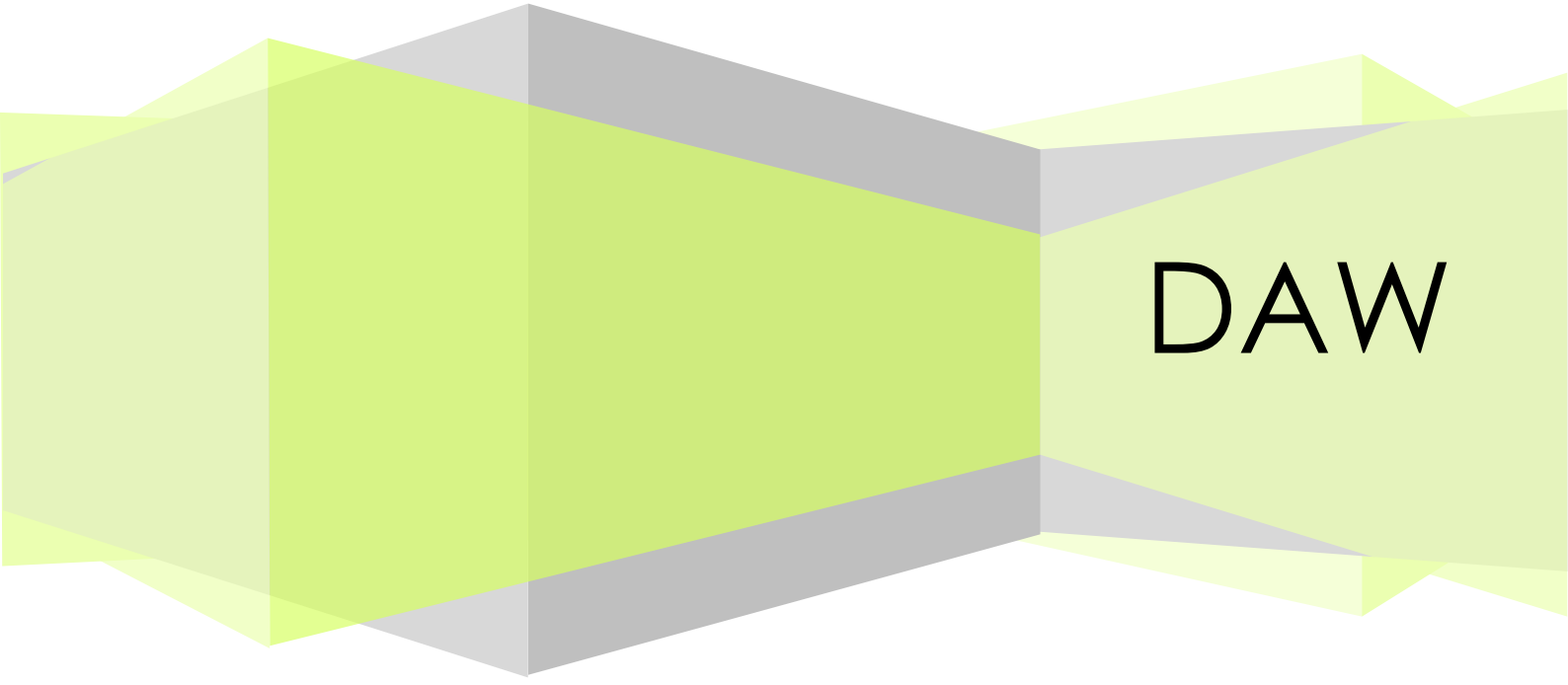
Departamento de informática y comunicaciones

Desarrollo web en entorno servidor

Manejo de errores

Unidad 6

Yolanda Iglesias Suárez



Contenido

UNIDAD 6. Manejo de errores

Al crear secuencias de comandos y aplicaciones web, gestión de errores es una parte importante. Si el código carece de código de comprobación de errores, el programa puede parecer muy poco profesional y que puede estar abierto a riesgos de seguridad.
Este tutorial contiene algunos de los errores más comunes control de los métodos en PHP.
Vamos a mostrar los diferentes métodos de tratamiento de errores:

- Sencillo " die() " declaraciones
- Errores personalizados y disparadores de error
- Error al reportar

Básicas de gestión de error: Uso de la die() función

El primer ejemplo muestra una secuencia de comandos simple que se abre un archivo de texto:

```
<?php
$file=fopen("welcome.txt","r");
?>
```

Si no existe el archivo podría obtener un error como este:

```
Warning : fopen(welcome.txt) [function.fopen]: failed to open stream:
No such file or directory in C:\webfolder\test.php on line 2
```

Para evitar que el usuario un mensaje de error similar a la anterior, probamos si el archivo existe antes de intentar acceder a ella:

```
<?php
if(!file_exists("welcome.txt")) {
  die("File not found");
} else {
  $file=fopen("welcome.txt","r");
}
?>
```

Ahora bien, si no existe el archivo se obtiene un error como este:

```
File not found
```

El código anterior es más eficiente que el código anterior, ya que utiliza un simple mecanismo de manejo de errores para detener el guión tras el error.

Sin embargo, sólo tiene que parar el guión no es siempre el camino correcto a seguir. Vamos a echar un vistazo a las funciones de PHP alternativas para el manejo de errores.

La creación de un controlador de errores personalizado

La creación de un controlador de errores personalizado es bastante simple. Sólo tendrá que crear una función especial que se puede llamar cuando se produce un error en PHP.

Esta función debe ser capaz de manejar un mínimo de dos parámetros (nivel de error y el mensaje de error), pero puede aceptar hasta cinco parámetros (optionally: file, line-number, and the error context) :

Sintaxis

error_function(error_level,error_message,
error_file,error_line,error_context)

Parámetro	Descripción
error_level	Necesario. Especifica el nivel de informe de error para el error definido por el usuario. Debe ser un número de valor. Consulte la tabla siguiente para los posibles niveles de informe de error
error_message	Necesario. Especifica el mensaje de error para el error definido por el usuario
error_file	Opcional. Especifica el nombre de archivo en el que se produjo el error
error_line	Opcional. Especifica el número de línea en el que se produjo el error
error_context	Opcional. Especifica una matriz que contiene todas las variables y sus valores, en uso cuando se produjo el error

Error niveles de informe

Estos niveles de informe de error son los diferentes tipos de error al controlador de error definido por el usuario se puede utilizar para:

Valor	Constante	Descripción
2	E_WARNING	No fatales errores de ejecución. La ejecución del script no se detiene
8	E_NOTICE	avisos en tiempo de ejecución. El script encontró algo que podría ser un error, pero también podría ocurrir cuando se ejecuta un script normalmente
256	E_USER_ERROR	Fatal error generado por el usuario. Esto es como un E_ERROR establecido por el programador utilizando la función de PHP trigger_error()
512	E_USER_WARNING	No fatal generado por los usuarios de advertencia. Esto es como un E_WARNING establecido por el programador utilizando la función de PHP trigger_error()
1024	E_USER_NOTICE	aviso generado por el usuario. Esto es como un E_NOTICE establecido por el programador utilizando la función de PHP trigger_error()
4096	E_RECOVERABLE_ERROR	capturable error fatal. Esto es como un E_ERROR pero puede ser atrapado por un mango definida por el usuario (see also set_error_handler())
8191	E_ALL	Todos los errores y advertencias (E_STRICT became a part of E_ALL in PHP 5.4)

Ahora vamos a crear una función para controlar los errores:

```
function customError($errno, $errstr) {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Ending Script";
  die();
}
```

El código anterior es una simple función de control de errores. Cuando se activa, se pone el nivel de error y un mensaje de error. A continuación, emite el nivel de error y el mensaje y termina el guión.

Ahora que hemos creado una función de control de errores tenemos que

decidir cuando debe ser activado.

Conjunto controlador de errores

El gestor de errores por defecto de PHP está construido en el gestor de errores. Vamos a hacer que la función anterior del controlador de errores por defecto para la duración de la secuencia de comandos.

Es posible cambiar el gestor de errores para solicitar sólo algunos errores, de esa manera la secuencia de comandos puede controlar errores diferentes en diferentes maneras. Sin embargo, en este ejemplo vamos a utilizar nuestro gestor de errores a medida para todos los errores:

```
set_error_handler("customError");
```

Como queremos que nuestra función personalizada para manejar todos los errores, la set_error_handler()sólo se necesita un parámetro, un segundo parámetro podría ser añadido para especificar un nivel de error.

Ejemplo

Probando el controlador de errores al tratar de variable de salida que no existe:

```
<?php
//error handler function
function customError($errno, $errstr) {
  echo "<b>Error:</b> [$errno] $errstr";
}

//set error handler
set_error_handler("customError");

//trigger error
echo($test);
?>
```

La salida del código anterior debería ser algo como esto:

```
Error: [8] Undefined variable: test
```

Desencadenar un error

En un guión donde los usuarios pueden introducir datos es útil para provocar errores cuando se produce una entrada ilegal. En PHP, esto se hace por el trigger_error() función.

Ejemplo

En este ejemplo se produce un error si el "test" variable es más grande que "1" :

```
<?php
$test=2;
if ($test>=1) {
  trigger_error("Value must be 1 or below");
}
?>
```

La salida del código anterior debería ser algo como esto:

```
Notice : Value must be 1 or below
in C:\webfolder\test.php on line 6
```

Un error puede ser activado en cualquier lugar que desee en un guión, y mediante la adición de un segundo parámetro, puede especificar lo que se activa el nivel de error.

tipos de errores posibles:

- E_USER_ERROR - Error en tiempo de ejecución generados por los usuarios fatal. Los errores que no pueden ser recuperados a partir. La ejecución del script se detiene
- E_USER_WARNING - no fatal generado por los usuarios de alerta en tiempo de ejecución. La ejecución del script no se detiene
- E_USER_NOTICE - por defecto. notificación en tiempo de ejecución generado por el usuario. El script encontró algo que podría ser un error, pero también podría ocurrir cuando se ejecuta un script normalmente

Ejemplo

En este ejemplo un E_USER_WARNING se produce si el "test" variable es más grande que "1" . Si se produce un E_USER_WARNING usaremos nuestro manipulador personalizado y poner fin a la secuencia de comandos:

```
<?php
//error handler function
function customError($errno, $errstr) {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Ending Script";
  die();
}

//set error handler
set_error_handler("customError",E_USER_WARNING);

//trigger error
$test=2;
```

```
if ($test>=1) {
  trigger_error("Value must be 1 or below",E_USER_WARNING);
}
?>
```

La salida del código anterior debería ser algo como esto:

```
Error: [512] Value must be 1 or below
Ending Script
```

Ahora que hemos aprendido a crear nuestros propios errores y cómo activarlas, permite echar un vistazo a el registro de errores.

Registro de errores

Por defecto, PHP envía un registro de errores de sistema de registro del servidor o un archivo, dependiendo de cómo la configuración error_log en el archivo php.ini. Al utilizar el error_log() la función que puede enviar registros de errores en un archivo especificado o un destino remoto.

El envío de mensajes de error a sí mismo por correo electrónico puede ser una buena manera de conseguir la notificación de errores específicos.

Enviar un mensaje de error por E-Mail

En el siguiente ejemplo vamos a enviar un e-mail con un mensaje de error y terminar el guión, si se produce un error específico:

```
<?php
//error handler function
function customError($errno, $errstr) {
  echo "<b>Error:</b> [$errno] $errstr<br>";
  echo "Webmaster has been notified";
  error_log("Error: [$errno] $errstr",1,
  "someone@example.com","From: webmaster@example.com");
}

//set error handler
set_error_handler("customError",E_USER_WARNING);

//trigger error
$test=2;
if ($test>=1) {
  trigger_error("Value must be 1 or below",E_USER_WARNING);
}
?>
```

La salida del código anterior debería ser algo como esto:

```
Error: [512] Value must be 1 or below
Webmaster has been notified
```

Y el correo recibido en el código anterior es el siguiente:

```
Error: [512] Value must be 1 or below
```

Esto no debe ser utilizado con todos los errores. Errores regulares deben ser registrados en el servidor mediante el sistema por defecto de PHP registro.