PROGRAMACIÓN WEB EN ENTORNO SERVIDOR UD-10

EXCEPCIONES-ERRORES

ÍNDICE

[VARIABLES DE SERVIDOR 3](#_Toc147867406)

[A. $\_SERVER[] 3](#_Toc147867407)

[VALIDACIÓN DE DATOS 4](#_Toc147867408)

[TEXTO IMPRIMIBLE 4](#_Toc147867409)

[INTRODUCCIÓN DE <br /> 4](#_Toc147867410)

[FORMATEO SALIDA CON PRINTF 5](#_Toc147867411)

[FORMATEO SALIDA CON NUMBER\_FORMAT 5](#_Toc147867412)

[FUNCIONES Y CONSTANTES MATEMÁTICAS 6](#_Toc147867413)

# ERRORES

PHP clasifica los errores que ocurren en diferentes niveles. Cada nivel se identifica con una constante. Por ejemplo:

* **E\_ERROR**: errores fatales, no recuperables. Se interrumpe el script.
* **E\_WARNING**: advertencias en tiempo de ejecución. El script no se interrumpe.
* **E\_NOTICE**: avisos en tiempo de ejecución.

Para la configuración de los errores podemos hacerlo de dos formas:

* A nivel de php.ini (en el propio fichero nos indican la configuración recomendable en desarrollo y producción):
  + error\_reporting: indica los niveles de errores a notificar
  + error\_reporting = E\_ALL & ~E\_NOTICE -> Todos los errores menos los avisos en tiempo de ejecución. (E\_ALL, E\_WARNING, E\_PARSE, E\_NOTICE , E\_STRICT, E\_DEPRECATED)
  + display\_errors: indica si mostrar o no los errores por pantalla. En entornos de producción es común ponerlo a off
* Mediante código con las siguientes funciones:
  + **error\_reporting(codigo)** 🡪 Controla qué errores notificar. Modifica en tiempo de ejecución la directiva error\_reporting.
  + **set\_error\_handler(nombreManejador)** 🡪 Indica que función se invocará cada vez que se encuentre un error. El manejador recibe como parámetros el nivel del error y el mensaje

# EXCEPCIONES

La gestión de excepciones en PHP es similar a Java, haciendo uso de un bloque try / catch / finally. Si detectamos una situación anómala y queremos lanzar una excepción, deberemos realizar throw new Exception (adjuntando el mensaje que lo ha provocado).

En PHP no se lanzan excepciones automáticamente, siempre hay que lanzar un throw.

Texto

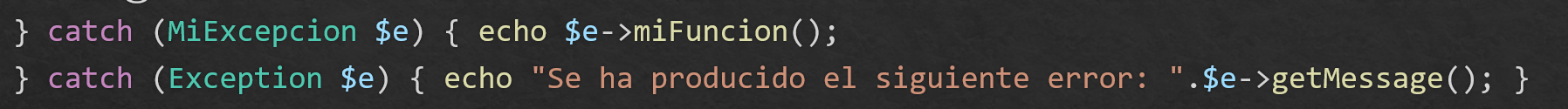
Descripción generada automáticamente

La clase Exception es la clase padre de todas las excepciones. Su constructor recibe mensaje[,codigoError][,excepcionPrevia].

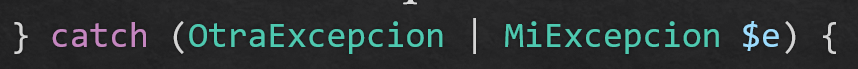
A partir de un objeto Exception, podemos acceder a los métodos getMessage()y getCode() para obtener el mensaje y el código de error de la excepción capturada.

El propio lenguaje ofrece un conjunto de excepciones ya definidas, las cuales podemos capturar y lanzar. Se recomienda su consulta en la documentación oficial.

Se pueden usar excepciones múltiples para comprobar diferentes condiciones. A la hora de capturarlas, se hace de más específica a más general colocando varios catch.



Si en el mismo catch queremos capturar varias excepciones, hemos de utilizar el operador tubería |.



Existe el tipo Throwable, el cual es un interfaz que implementan tanto los errores como las excepciones, y nos permite capturar los dos tipos a la vez.

Un letrero de color negro

Descripción generada automáticamente con confianza media

Si sólo queremos capturar los errores fatales, podemos hacer uso de la clase Error en el catch.

Imagen que contiene Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

# CLASE EXCEPTION

La clase Exception dispone de los siguientes métodos:

* getMessage( ): Obtiene el mensaje de Excepción.
* getPrevious( ): Devuelve la excepción anterior..
* getCode( ): Obtiene el código de una excepción
* getFile( ): Obtiene el fichero en el que se creó la excepción.
* getLine( ): Obtiene la línea en el que se creó la excepción.
* getTrace( ): Obtiene una matriz, que indica donde se ha producido la excepción.
* getTraceAsString( ): Obtiene la traza de la pila como una cadena de caracteres.
* \_\_toString( ): Representación de la excepción en formato cadena. Este método es posible sobrecargarlo y es el que utilizaremos para definir nuevas clases de excepción.

# TIPOS DE ERRORES Y EXCEPCIONES CAPTURABLES

Tabla

Descripción generada automáticamente