PHP Error Handling

Topics : PHP
Written on February 04, 2021

Error handling is the process of catching errors occured by your program and then taking appropriate action. PHP system provides many error handling methods to manage errors occured during execution time. We can prevent stopping execution of program if error occured during execution of code using Error handling procees in php.

There are various different error handling methods as below.

1. Using die() statements
2. Defining Custom Error Handling Function
3. Error reporting

Using die() statements

While writing your PHP program you should check all possible error condition before going ahead and take appropriate action when required.

PHP Example :

```
<?php
   if(file_exists("aryatechno.txt")) {
     $file = fopen("aryatechno.txt", "r");
   } else {
     die("Error: The file does not exist!");
   }
 ?>
```

Output:

Error: The file does not exist!

Explanation : Above program checks file is available or not. if file is not available , then it display error message mentioned in die() function.

Defining Custom Error Handling Function

We can create simple custom function to handle error in program.

PHP Syntax:

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

Parameter,

error_level : Required - Specifies the error report level for the user-defined error. Must be a value number. See table below for possible error report levels
error_message : Required - Specifies the error message for the user-defined error
error_file : Optional - Specifies the filename in which the error occurred
error_line : Optional - Specifies the line number in which the error occurred
error_context : Optional - Specifies an array containing every variable, and their values, in use when the error occurred

PHP error constants or levels for Error reporting.

These error report levels are the different types of error which are generated during execution of code.

1. E_ERROR : A fatal error that causes script termination
2. E_WARNING : Run-time warning that does not cause script termination
3. E_PARSE : Compile time parse error.
4. E_NOTICE : Run time notice caused due to error in code
5. E_CORE_ERROR : Fatal errors that occur during PHP's initial startup (installation)
6. E_DEPRECATED : Run-time notices.
7. E_COMPILE_ERROR : Fatal compile-time errors indication problem with script.
8. E_RECOVERABLE_ERROR : Catchable fatal error indicating a dangerous error
9. E_USER_ERROR : User-generated error message.
10. E_USER_WARNING : User-generated warning message.
11. E_USER_NOTICE : User-generated notice message.
12. E_STRICT : Run-time notices.

All the above error level can be set using PHP built-in library function where level can be any of the value defined in above constants.

error_reporting () function is used to allow to display error message for different error types in web browser using above error level. error_reporting () function is used to manage showing different error types in web browser. You can show or hide error message on web browser using error_reporting () function when error occured in program.

PHP Syntax:

int error_reporting ([int $level])

i.e error_reporting(E_ERROR); only displays fatal error.

Set Error Handler

Set Error Handler function is used to call created custom error handler function which display error in web browser.

PHP Syntax:

```
set_error_handler("customErrorFunction");
```

PHP Example :

```
<?php
//Create error handler function
function customErrorFunc($errno, $errstr) {
  echo "Error handle: [$errno] $errstr";
}

//set error handler to call error handler function customErrorFunc
set_error_handler("customErrorFunc");

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

Output:

```
Error handle: [2] Undefined variable $str
```

Trigger an Error

trigger_error() function is used to display custom error in web browser when an illegal input occurs.

PHP Syntax:

```
trigger_error("custom message");
```

PHP Example :

```
<?php
$username="world212";
if (strlen($username)<9) {
  trigger_error("Username must be more than 8 character!");
}
?>
```

Output:

```
PHP Notice:  Username must be more than 8 character! in
C:\xampp\htdocs\phptraining\register.php on line 5
```

Output :