

CSCI 241

Handling Errors

Department of CS & QM
Winthrop University
 Fall 2011

Prepared by X. Wang

Topics

- PHP Errors
- Error Reporting Configuration
- Custom Error Handlers
- Apache Web Server errors

Prepared by X. Wang

2

PHP Errors

- No matter what languages we are using, there are three general types of program errors: **syntax errors**, **runtime errors**, and **logic errors**
- In PHP scripts, we may get these errors too. We need to find some general ways to detect, handle, avoid and/or solve those errors

Prepared by X. Wang

3

PHP Errors – Syntax Errors

- **Syntax errors**: the PHP script doesn't follow the rules of PHP language
- Syntax errors are detected at interpreting time. So it is also called **Parser errors**
- When the PHP parser cannot parse PHP statements, the parser will show some error messages
- General syntax error examples: missed semicolons, quotations not paired, missed comma when passing parameters to a function, and so on

Prepared by X. Wang

4

PHP Errors – Syntax Errors

- The following PHP code

```
<?php
  print This is the first page";
?>
```

produces the error message as

Parse error: parse error, unexpected T_STRING in `/home/xwang/public_html/csci241/code/test1.php` on line 2

Prepared by X. Wang

5

PHP Errors – Runtime Errors

- **Runtime errors**: runtime errors are not caused solely by the contents of your script. They can rely on interactions between your scripts and other conditions
- Runtime errors can be harder to detect and fix
- For example, the following PHP code is valid, and no errors

```
<?php
  require "db.php";
?>
```

Prepared by X. Wang

6

PHP Errors – Runtime Errors

- But if the `db.php` file is not saved in the same folder with the PHP page, you will see the following error message in your web browser

Fatal error: main(): Failed opening required 'db.php' (include_path='.:usr/share/pear') in `/home/xwang/public_html/csci241/code/test2.php` on line 2

Prepared by X. Wang

7

PHP Errors – Logic Errors

- **Logic errors:** the code itself has no problem, but the programmer didn't use a right logic to implement the code
- Logic errors can be the hardest type of error to find and eliminate
- Since the code has no problem, there is no error messages to be reported by the system
- A new programmer must do more practices on coding to reduce logic errors

Prepared by X. Wang

8

PHP Errors – Logic Errors

- In the following `loop` statement, a programmer wants to print the string "Call me immediately" 5 times, but the code was written as

```
<?php
for ($i=0; $i<5; $i++);
    print "Call me immediately.";
?>
```

- Because of the additional `;` at the end of the first line, this `loop` statement will **print the string only once**
- The above code has no syntax problems

Prepared by X. Wang

9

PHP Errors

- Now let's see how to handle errors in PHP engine and Web server (Apache), and show how error reporting can be configured for debugging during development
- When we write PHP scripts, the code may create errors or problems

Prepared by X. Wang

10

PHP Errors

- Based on the errors we discussed previously, we can classify PHP errors in four different situations:
 - internally within the PHP core (system error)
 - during compilation when the code is first loaded (parser error)
 - at run time when the code is being executed (runtime error)
 - explicitly triggered by the user code (user error)

Prepared by X. Wang

11

PHP Errors

- For each situation, we can break PHP errors down to three types or levels: errors, warnings, and notices
- The following are the error codes (values) and constants defined for error reporting in PHP

Prepared by X. Wang

12

PHP Errors

Value	Constant	Description	Halts script?
1	E_ERROR	Fatal runtime error	Yes
2	E_WARNING	Non-fatal runtime errors	No
4	E_PARSE	Compile-time parse errors	Yes
8	E_NOTICE	Non-fatal run-time notices	No
16	E_CORE_ERROR	Fatal PHP startup errors	Yes
32	E_CORE_WARNING	Non-fatal PHP startup errors	No
64	E_COMPILE_ERROR	Fatal compile-time errors	Yes
128	E_COMPILE_WARNING	Non-fatal compile-time errors	No
256	E_USER_ERROR	Fatal user-generated errors	Yes
512	E_USER_WARNING	Non-fatal user-generated errors	No
1024	E_USER_NOTICE	User-generated notices	No
2048	E_STRICT	(Added from 5.0.0)	No
4096	E_RECOVERABLE_ERROR	(Added from 5.2.0)	No
6143	E_ALL	All of the above (6143 after 6.0.0)	No

Prepared by X. Wang

13

PHP Errors

- The **ERROR** type includes errors such as calling undefined functions, instantiating objects of a non-existing class, and issuing a statement where it is not allowed
- The **ERROR** type will **stop script execution**. So the script produces output only to the points where the error occurred
- The **E_PARSE** is one kind **ERROR** type that includes syntax errors from missing semicolons, missing quotes and brackets, and statements with incorrect numbers of parameters

Prepared by X. Wang

14

PHP Errors

- The **WARNING** type covers less serious problems, where a script may continue successfully, such as most of MySQL related errors
- The **WARNING** type **does not stop script execution**
- The **NOTICE** type is usually minor and informational. It does not stop script execution

Prepared by X. Wang

15

PHP Errors

- By default, error messages are generated by **PHP engine**, and displayed to the user's browser
- Based on the error messages, the **programmer should fix all the errors** during development time
- An error message usually shows the **error type**, **error text message**, the **source page file** and the **line number** where the code has the error
- The following example shows a **WARNING** message. The PHP statements after the error are still executed

Prepared by X. Wang

16

PHP Errors

- The following is a PHP file ([errortest.php](#)). It has a problem

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>Error Test</title>
</head>
<body>
<h1>Two times!</h1>
<?php
function double($num) {
    return $num * 2;
}
print "Two times ten is: ", double();
print "<br />Two times five is: ", double(5);
?>
</body>
</html>
```

Prepared by X. Wang

17

PHP Errors

- When we access the page, we got the following display
- It shows there is a **WARNING** message

```
Two times!
Warning: Missing argument 1 for double() in /home/xwang/public_html/csc241/code/errortest.php on line 8
Two times ten is: 0
Two times five is: 10
```

Prepared by X. Wang

18

PHP Errors

- During development time, the error messages produced by PHP are useful tools for debugging the code
- For example, from the error message in the previous example, we can see:
 - The error is a WARNING type
 - The reason is missing a parameter when calling double()
 - The error occurs in the `errortest.php` file, and
 - On the line 10 in the file

Prepared by X. Wang

19

PHP Code Debugging

- The following is another example, we call the predefined `mysql_connect()` function, but didn't provide a correct server name, so the call is failure
- When we access this page, it displays the following WARNING message

Connect to MySQL:

```
Warning: mysql_connect(): Unknown MySQL server host 'local' (2)
in /home/xwang/public_html/csc241/code/errortest2.php on line 8
Could not connect
```

Prepared by X. Wang

20

PHP Code Debugging

- Another error example (`errortest2.php`)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>Error Test</title></head>
<body>
<h2>Connect to MySQL:</h2>
<?php
    if (!($connection = mysql_connect("local")))
        die("Could not connect");
    if (!(@ mysql_select_db($dbname, $connection)))
        showerror();
?>
</body>
</html>
```

Prepared by X. Wang

21

Error Reporting Configuration

- Errors provide useful information for programmers to debug the application
- However, when it is deployed, the PHP error messages displayed in the application output is **messy**, **confusing** for users, and **uninformative** for those who need to be alerted to rectify the problems
- Most importantly, it is also a **security problem**: program internals are displayed as part of error messages and these shouldn't be displayed to end users

Prepared by X. Wang

22

Error Reporting Configuration

- So we need to have a way to control the error reporting
- Error reporting is configured in PHP in two ways
- First, we can set the global variable in the `php.ini` file to control the error reporting levels
- By default, in the `php.ini` file, the error reporting is set to `error_reporting = E_ALL`
- To allow error messages be displayed in web pages, the following control needs to be set to on `display_errors = on`

Prepared by X. Wang

23

Error Reporting Configuration

- If we want to ignore all warnings and notices, we can set


```
error_reporting = E_ERROR | E_CORE_ERROR
                | E_COMPILE_ERROR | E_PARSE
```
- The settings in `php.ini` file controls error reporting for all websites and pages deployed on the web server

Prepared by X. Wang

24

Error Reporting Configuration

- **Second**, we can control the error reporting in PHP script by calling the `error_reporting()` function
- This control level is only for the current PHP page
- For example, setting the error reporting to all, we can call `error_reporting(E_ALL);`
- Calling this function without a parameter will return the current setting in the PHP engine

Prepared by X. Wang

25

Error Reporting Configuration

- Usually, during development stage, we turn on all error reporting using the global `php.ini` setting
- However, we don't recommend to use this `E_ALL` setting for the application deployment because we don't want to show the program internals to the end user

Prepared by X. Wang

26

Custom Error Handlers

- We can also define custom error handlers
- We discuss how to **add a professional error handler** to an application, and how to improve the internal PHP error handlers to produce even more information during development
- The `set_error_handler()` function allows us to define a custom error handler that replaces the internal PHP error handler for **non-critical errors**

Prepared by X. Wang

27

Custom Error Handlers

- `string set_error_handler(string error_handler)`
- This function takes one parameter, a user-defined `error_handler` function that is called whenever a non-critical error occurs
- The custom error handler is not called for the following errors: `E_ERROR`, `E_PARSE`, `E_COMPILE_ERROR`, `E_COMPILE_WARNING`, `E_CORE_ERROR`, `E_CORE_WARNING`, and most of `E_STRICT`
- For those error types, the PHP internal error handler is always used

Prepared by X. Wang

28

Custom Error Handlers

- A custom error handler function must accept at least two parameters: an **integer error number** and a **descriptive error string**
- Three additional optional parameters can also be used:
 - A string representing the **filename** of the script that caused the error
 - An **integer line number** indicating the line in the file where the error was noticed
 - An array of **additional variable context information**

Prepared by X. Wang

29

Custom Error Handlers

- In the following example, we use the **custom error handler** to process the warning error presented the "Two times!" example previously
- To set up a new custom error handler that is defined in the function `customHandler()`, we register it with `set_error_handler("customHandler");`
- **Notice**: it doesn't include the **parentheses** of the function as part of the parameter string

Prepared by X. Wang

30

Custom Error Handlers

- This example is saved as `errortest3.php`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>Error Test 3</title></head>
<body><h2>Two times!</h2>
<?php
function customHandler($number, $message, $file, $line, $context) {
    if ($number==E_WARNING || $number==E_NOTICE) {
        print "<hr /><b>Custom Error Handler -- Warning/Notice</b><br />";
        print "An error has occurred on ($line) in the ($file) file.<br />";
        print "The error is a '$message'" (error #($number))-<br />";
        print "Here is the context information:<pre>";
        print_r($context);
        print "</pre><br />";
    }
}
```

Prepared by X. Wang

31

Custom Error Handlers

```
function double($num) {
    return $num * 2;
}

set_error_handler("customHandler");

print "Two times ten is: " . double();
print "<br />Two times five is: " . double(5);

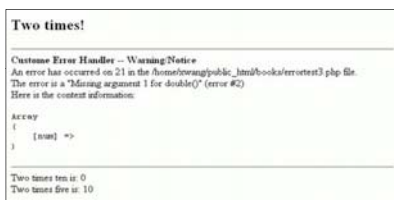
?>
</body>
</html>
```

Prepared by X. Wang

32

Custom Error Handlers

- When we access the file, it presents the following page



Prepared by X. Wang

33

Apache Web Server Errors

- As we know, when an HTTP request is sent to the web server, we may get some errors from the server.
- For example:
 - 400: Bad Request
 - 401: Authorization Required
 - 403: Forbidden
 - 404: Not Found
 - 500: Internal Server Error

Prepared by X. Wang

34

Apache Web Server Errors

- If we want to replace the (internal) error messages with customized messages, we can configure the Apache configuration file `httpd.conf`
- Using the `ErrorDocument` directive in `httpd.conf` file, we can link an external PHP page to display a custom message for an HTTP error

Prepared by X. Wang

35

Apache Web Server Errors

- For example, adding the following directives to the `httpd.conf` file will direct the errors 400, 401, 403, 404 and 500 to the PHP page `error.php`

```
ErrorDocument 400 "/error.php?400"
ErrorDocument 401 "/error.php?401"
ErrorDocument 403 "/error.php?403"
ErrorDocument 404 "/error.php?404"
ErrorDocument 500 "/error.php?500"
```

- Note: after these changes are made in `httpd.conf` file, we need to restart Apache web server

Prepared by X. Wang

36

Apache Web Server Errors

- The following is the partial code for `error.php` to show how to use PHP code to generate the customized error messages
- To match with the `ErrorDocument` directives on the previous page, we need to save `error.php` into Apache's document root folder `htdocs` on Windows
- Please notice how to fetch the query string from a URL through the global associative array variable `$_SERVER`

Prepared by X. Wang

37

Apache Web Server Errors

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>Error Page</title>
</head>
<body>
<?php
$error_no = $_SERVER("QUERY_STRING");

switch ($error_no) {
case 400:
$error_output = "<h1>\\"Bad Request\\" Error Page - (Error Code 400)</h1>";
$error_output .= "The browser has made a Bad request<br />";
$error_output .= "<a href='mailto:sysadmin@localhost.com'>Contact</a>";
$error_output .= "the system administrator if you feel this to be in error";
break;
.....
```

Prepared by X. Wang

38

Apache Web Server Errors

```
case 404:
$error_output = "<h1>\\"Page NOT Found\\" Error Page - (Error Code 404)</h1>";
$error_output .= "The page you are looking for cannot be found.<br />";
$error_output .= "<a href='mailto:sysadmin@localhost.com'>Contact</a>";
$error_output .= "the system administrator if you feel this to be in error";
break;
default:
$error_output = "<h2>Error Page - (Error Code ($error_no))</h2>";
}
echo $error_output;
?>
</body>
</html>
```

Prepared by X. Wang

39

Apache Web Server Errors

- The following is the new error message after accessing a non-existing page on the server



Prepared by X. Wang

40