**在 PHP 中，默认的错误处理很简单。一条消息会被发送到浏览器，这条消息带有文件名、行号以及一条描述错误的消息。**

**PHP 错误处理**

在创建脚本和 web 应用程序时，错误处理是一个重要的部分。如果您的代码缺少错误检测编码，那么程序看上去很不专业，也为安全风险敞开了大门。

本教程介绍了 PHP 中一些最为重要的错误检测方法。

我们将为您讲解不同的错误处理方法：

* 简单的 "die()" 语句
* 自定义错误和错误触发器
* 错误报告

**基本的错误处理：使用 die() 函数**

第一个例子展示了一个打开文本文件的简单脚本：

<?php

$file=fopen("welcome.txt","r");

?>

如果文件不存在，您会获得类似这样的错误：

**Warning**: fopen(welcome.txt) [function.fopen]: failed to open stream:

No such file or directory in **C:\webfolder\test.php** on line **2**

为了避免用户获得类似上面的错误消息，我们在访问文件之前检测该文件是否存在：

<?php

if(!file\_exists("welcome.txt"))

{

die("File not found");

}

else

{

$file=fopen("welcome.txt","r");

}

?>

现在，假如文件不存在，您会得到类似这样的错误消息：

File not found

比起之前的代码，上面的代码更有效，这是由于它采用了一个简单的错误处理机制在错误之后终止了脚本。

不过，简单地终止脚本并不总是恰当的方式。让我们研究一下用于处理错误的备选的 PHP 函数。

**创建自定义错误处理器**

创建一个自定义的错误处理器非常简单。我们很简单地创建了一个专用函数，可以在 PHP 中发生错误时调用该函数。

该函数必须有能力处理至少两个参数 (error level 和 error message)，但是可以接受最多五个参数（可选的：file, line-number 以及 error context）：

**语法**

error\_function(error\_level,error\_message,

error\_file,error\_line,error\_context)

|  |  |
| --- | --- |
| **参数** | **描述** |
| **error\_level** | **必需。为用户定义的错误规定错误报告级别。必须是一个值数。**  **参见下面的表格：错误报告级别。** |
| **error\_message** | **必需。为用户定义的错误规定错误消息。** |
| **error\_file** | **可选。规定错误在其中发生的文件名。** |
| **error\_line** | **可选。规定错误发生的行号。** |
| **error\_context** | **可选。规定一个数组，包含了当错误发生时在用的每个变量以及它们的值。** |

**错误报告级别**

**这些错误报告级别是错误处理程序旨在处理的错误的不同的类型：**

|  |  |  |
| --- | --- | --- |
| **值** | **常量** | **描述** |
| **2** | **E\_WARNING** | **非致命的 run-time 错误。不暂停脚本执行。** |
| **8** | **E\_NOTICE** | **Run-time 通知。**  **脚本发现可能有错误发生，但也可能在脚本正常运行时发生。** |
| **256** | **E\_USER\_ERROR** | **致命的用户生成的错误。这类似于程序员使用 PHP 函数 trigger\_error() 设置的 E\_ERROR。** |
| **512** | **E\_USER\_WARNING** | **非致命的用户生成的警告。这类似于程序员使用 PHP 函数 trigger\_error() 设置的 E\_WARNING。** |
| **1024** | **E\_USER\_NOTICE** | **用户生成的通知。这类似于程序员使用 PHP 函数 trigger\_error() 设置的 E\_NOTICE。** |
| **4096** | **E\_RECOVERABLE\_ERROR** | **可捕获的致命错误。类似 E\_ERROR，但可被用户定义的处理程序捕获。(参见 set\_error\_handler())** |
| **8191** | **E\_ALL** | **所有错误和警告，除级别 E\_STRICT 以外。**  **（在 PHP 6.0，E\_STRICT 是 E\_ALL 的一部分）** |

现在，让我们创建一个处理错误的函数：

function customError($errno, $errstr)

{

echo "<b>Error:</b> [$errno] $errstr<br />";

echo "Ending Script";

die();

}

上面的代码是一个简单的错误处理函数。当它被触发时，它会取得错误级别和错误消息。然后它会输出错误级别和消息，并终止脚本。

现在，我们已经创建了一个错误处理函数，我们需要确定在何时触发该函数。

## Set Error Handler

PHP 的默认错误处理程序是内建的错误处理程序。我们打算把上面的函数改造为脚本运行期间的默认错误处理程序。

可以修改错误处理程序，使其仅应用到某些错误，这样脚本就可以不同的方式来处理不同的错误。不过，在本例中，我们打算针对所有错误来使用我们的自定义错误处理程序：

set\_error\_handler("customError");

由于我们希望我们的自定义函数来处理所有错误，set\_error\_handler() 仅需要一个参数，可以添加第二个参数来规定错误级别。

### 实例

通过尝试输出不存在的变量，来测试这个错误处理程序：

<?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);

?>

以上代码的输出应该类似这样：

Error: [8] Undefined variable: test

**触发错误**

在脚本中用户输入数据的位置，当用户的输入无效时触发错误的很有用的。在 PHP 中，这个任务由 trigger\_error() 完成。

**例子**

在本例中，如果 "test" 变量大于 "1"，就会发生错误：

<?php

$test=2;

if ($test>1)

{

trigger\_error("Value must be 1 or below");

}

?>

以上代码的输出应该类似这样：

Notice: Value must be 1 or below

in C:\webfolder\test.php on line 6

您可以在脚本中任何位置触发错误，通过添加的第二个参数，您能够规定所触发的错误级别。

**可能的错误类型：**

* E\_USER\_ERROR - 致命的用户生成的 run-time 错误。错误无法恢复。脚本执行被中断。
* E\_USER\_WARNING - 非致命的用户生成的 run-time 警告。脚本执行不被中断。
* E\_USER\_NOTICE - 默认。用户生成的 run-time 通知。脚本发现了可能的错误，也有可能在脚本运行正常时发生。

**例子**

在本例中，如果 "test" 变量大于 "1"，则发生 E\_USER\_WARNING 错误。如果发生了 E\_USER\_WARNING，我们将使用我们的自定义错误处理程序并结束脚本：

<?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);

}

?>

以上代码的输出应该类似这样：

Error: [512] Value must be 1 or below

Ending Script

现在，我们已经学习了如何创建自己的 error，以及如何触发它们，现在我们研究一下错误记录。

**错误记录**

默认地，根据在 php.ini 中的 error\_log 配置，PHP 向服务器的错误记录系统或文件发送错误记录。通过使用 error\_log() 函数，您可以向指定的文件或远程目的地发送错误记录。

通过电子邮件向您自己发送错误消息，是一种获得指定错误的通知的好办法。

### 通过 E-Mail 发送错误消息

在下面的例子中，如果特定的错误发生，我们将发送带有错误消息的电子邮件，并结束脚本：

<?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);

}

?>

以上代码的输出应该类似这样：

Error: [512] Value must be 1 or below

Webmaster has been notified

接收自以上代码的邮件类似这样：

Error: [512] Value must be 1 or below

这个方法不适合所有的错误。常规错误应当通过使用默认的 PHP 记录系统在服务器上进行记录。