

Exceptions – when and why?

Source: <http://coding.derkeiler.com/Archive/PHP/php.general/2008-12/msg00174.html>

- *From:* lawpoop <lawpoop@xxxxxxxxxx>
 - *Date:* Thu, 4 Dec 2008 07:30:02 -0800 (PST)
-

I'm working on a class where I might want to throw some exceptions. Now I'm wondering they why's and whens of exception throwing. I have several exceptions that could happen on __construct, some of which are more like warnings, or state notifications. Is this the proper use of an exception?

I guess my question is, what does an exception communicate to the client programmer? Does throwing an exception mean that something has gone horribly wrong, and the object is thereby unusable? The one thing I do know about exception behavior is that if they aren't caught, then the client programmer gets warnings popping up that they don't want.

In a lot of examples I found, they die on exception. I got the following code from google code, and here they die on an exception. It looks like they do so because the whole rest of the page hinges on the usage of the object.

```
<?php
require_once 'PDB.php';

try {
$db = PDB::connect('mysql:host=192.168.0.10;dbname=mydb',
'myname', 'secret');
$db->setFetchMode(PDO::FETCH_OBJ);
} catch (PDB_Exception $e) {

die("Could not connect: " . $e->getMessage() . "\n");

}

$sql = 'SELECT U.username FROM friends AS F JOIN users AS U ON
F.friendid = U.userid WHERE F.userid = ?';
$friends = $db->getCol($sql, 0, array((int)$_GET['userid']));
echo "Your friends are " . implode(', ', $friends);

?>
```

Or, can you use exceptions as warnings, or state notifications, with

Exceptions – when and why?

guaranteed delivery? In other words, I could have variables in my object that a client programmer could use to understand what has happened during instantiation, like:

```
if ( $myObj->state == "ok" ) {  
... object using code goes here ...  
  
}
```

but they may not think to check those variables. I might want to make sure that the client programmer has to catch the exception, so I know that they got the message. But, you can always say, it's up to the client programming to know how to use the object.

I have an class that I want to instantiate in the middle of a page. If it throws an exception, I really can't use it, so I don't need to execute any of the follow-up code that interacts with the object. But I also don't want to die in the middle of the page — I have the whole rest of the page to construct. So it seems I have to use some inelegant code to do so:

```
<?php  
  
... whole bunch of stuff ...  
  
$use_object = TRUE;  
  
try {  
$myObj =& new objClass();  
} catch (Exception $e) {  
  
echo 'Caught exception: ', $e->getMessage(), "\n";  
$use_object = FALSE;  
  
}  
  
if ( $use_object ) {  
... object code goes here ...  
  
}  
  
... rest of code ...  
  
?>  
.
```