

Re: 'undefined function' error if I use a fully-qualified include

Source: <http://coding.derkeiler.com/Archive/PHP/comp.lang.php/2005-10/msg00967.html>

- *From:* "Oli Filth" <catch@xxxxxxxxxxxxxxxxx>
 - *Date:* 19 Oct 2005 05:55:09 -0700
-

Puzzled wrote:

> Oli Filth wrote:

>

>> Puzzled said the following on 18/10/2005 17:54:

>>> If I use a fully qualified include call

>>>

>>> include ('<http://localhost/subtree/filename.php>')

>>>

>>> I get an 'undefined function' error when calling a routine that's

>>> defined in that file.

>>

>> This is an absolute *HTTP* URL, not an absolute *filesystem* URL. So it

>> obtains filename.php by performing an HTTP request to the specified

>> server, which just happens to be "localhost" in this case.

>>

>> And of course, a server set up to parse and execute PHP files will parse

>> and execute filename.php when it's requested, so all your include() call

>> sees is the output result of filename.php.

>

> I think I must be missing something important still, because I don't

> understand what you said. Check me on this, would you?

>

> A URL is just another filesystem reference, but using a highly

> generalized syntax that's independent of o/s and physical

> location--like NFS, but more general. So any processing that would

> be done or not done to a file would be independent of the type of

> reference used to retrieve it.

In this case, not really, because the URL is prefixed with "<http://>,
which specifies "Go and get this resource using the HTTP protocol".

The fact is that you're requesting "filename.php" via HTTP. PHP doesn't
know that you're expecting it to implicitly convert
"http://localhost/..." to a local filesystem reference, so it simply
sends out an HTTP request for that file, just as if you'd typed
"http://localhost/.../filename.php" into your browser.

The request just so happens to be to your own server, which serves this

Re: 'undefined function' error if I use a fully-qualified include

request by processing filename.php spitting the output back as the HTTP response. This HTTP response is then included into the original PHP file.

> When the php interpreter knows it's operating on an include file, it
> should know enough not to start a new symbol table, but instead add
> things like function definitions to the existing symbol table it
> started building when it read up the outermost file.

This is impossible, because filename.php is processed in a completely separate instance to the original PHP file, serving an entirely separate HTTP request.

> Is that what's causing the problem, the fact that I'm using a .php
> extension on the include file?
>
> That would make a crazy sort of sense, since the php interpreter would
> do its own filesystem fetches, but might hand any url reference, even
> one for an include, off to Apache to satisfy. And somewhere in that
> process, the fact that it's just an include file is getting pushed and
> the file is being interpreted as though it were a completely new job.
> Whereas if it had some .inc or .foo extension, Apache wouldn't try to
> pass it through the interpreter, but would just hand it over.

Yup, that's the explanation in a nutshell (except that the first PHP instance doesn't request filename.php directly from Apache, it does it via an HTTP request which lands at Apache indirectly).

The simplest solution would be to not include() local files via HTTP.

Even if you renamed your include files to .inc extensions, this would be insecure, since anyone could request them via HTTP and view your source code.

==
Oli

.

• **References:**

- ◆ ***Re: 'undefined function' error if I use a fully-qualified include***
◇ *From: Oli Filth*

- Prev by Date: ***Re: Professional Applications Devolpment***
- Next by Date: ***Re: 'undefined function' error if I use a fully-qualified include***
- Previous by thread: ***Re: 'undefined function' error if I use a fully-qualified include***
- Next by thread: ***Re: 'undefined function' error if I use a fully-qualified include***
- Index(es):
 - ◆ *Date*

Re: 'undefined function' error if I use a fully-qualified include

Re: 'undefined function' error if I use a fully-qualified include

◆ *Thread*