

Forum:[Hack Forums](#)

Thread: [Parse inbetween quotes.](#)

DJSanderrr - February 27, 2016, 8:58 am

Hello,

I was wondering what the fastest method is to parse the text inbetween quotes.

So that

Quote:"abc cba"

will become

Quote:abc cba

I've read many methods with regex, but I need to have about 500 of these requests a second, so was wondering if there's anything faster.

Thanks in advance.

Subayaku Kokoro - February 27, 2016, 9:05 am

what exactly are you trying to do?

do you only get 1 string with quotes ? or a whole text or whatever?

why just dont use

PHP Code

```
<?php
```

```
str_replace("'", "$string");
```

IF you just get a string

or if you really need regext make it like this

PHP Code

```
<?php
```

```
preg_match('/".*?"/', $x, $matches);
```

```
print_r($matches);
```

DJSanderrr - February 27, 2016, 9:22 am

(Feb 27, 2016 10:05 AM)KonVex Wrote: [\[CITE\]](#) why just dont use

PHP Code

```
<?php
```

```
str_replace("'", "$string");
```

Wow I was so stupid.. damn I really didn't think of that..

Thanks!

+Haze - February 27, 2016, 9:23 am

but if you have,

"my name" is "haze"
string replace won't work.

depending on the format you expect the data in, string replace might not be accurate.

Subayaku Kokoro - February 27, 2016, 10:53 am

(Feb 27, 2016 10:23 AM)+Haze Wrote: [\[CITE\]](#) but if you have,

"my name" is "haze"
string replace won't work.

depending on the format you expect the data in, string replace might not be accurate.
yeah thats why i asked what he is trying to do .. but i think he dont need any "complicated" solution for his "problem" ?

DJSanderrr - February 27, 2016, 10:56 am

(Feb 27, 2016 11:53 AM)KonVex Wrote: [\[CITE\]](#) yeah thats why i asked what he is trying to do .. but i think he dont need any "complicated" solution for his "problem" ?

Actually, I'm currently running against that problem..

I actually now need to only replace " to null if it's eighter at the beginning or the end of the string.

How would I do this WITHOUT regex, and first check if the character IS there.

+Haze - February 27, 2016, 11:03 am

Just use regex, it's probably your best bet. Here's the expression.

"(.*)"

I mean you can also use like str pos, but like that's just extra long code. Regex would suffice.

Haxalot - February 27, 2016, 11:09 am

Well, one way would be to set the first and last offset of the string to empty strings:

PHP Code

```
<?php
$a = "abc";
$a[0] = "";
$a[strlen($a) - 1] = "";
var_dump($a); // string(5) "abc"
```

Unfortunately though, this will set the empty string offsets to null byte characters (`\0`), hence why the string's length remains 5 characters long (I believe the bug has been fixed in PHP's master branch). Since the null byte characters may have adverse affects on your script, using [substr](#) would probably be the better approach:

PHP Code

```
<?php
$a = "abc";
var_dump(substr($a, 1, -1)); // string(3) "abc"
```

Obviously if the leading and trailing quote is optional, then you'll want to place in some checks.

DJSanderrr - February 29, 2016, 3:50 am

(Feb 27, 2016 12:09 PM)Haxalot Wrote: [\[CITE\]](#) Well, one way would be to set the first and last offset of the string to empty strings:

PHP Code

```
<?php
$a = "abc";
$a[0] = "";
$a[strlen($a) - 1] = "";
var_dump($a); // string(5) "abc"
```

Unfortunately though, this will set the empty string offsets to null byte characters (`\0`), hence why the string's length remains 5 characters long (I believe the bug has been fixed in PHP's master branch). Since the null byte characters may have adverse affects on your script, using [substr](#) would probably be the better approach:

PHP Code

```
<?php
$a = "abc";
var_dump(substr($a, 1, -1)); // string(3) "abc"
```

Obviously if the leading and trailing quote is optional, then you'll want to place in some checks. Thank you. I combined both your solutions somehow and works like a charm now!

Sakuto - February 29, 2016, 3:55 am

Strange that nobody suggested the trim function. It's perfect here. `trim($yourVariable, "");`

DJSanderrr - February 29, 2016, 6:54 am

(Feb 29, 2016 04:55 AM)Sakuto Wrote: [\[CITE\]](#) Strange that nobody suggested the trim function. It's perfect here. `trim($yourVariable, "");`

Just looked at that and seems awesome! Thank you.

Is it also sincificant faster than the other functions? I can't run a test at this moment :/

Sakuto - February 29, 2016, 7:14 am

I did not run any benchmark, but I highly doubt you're developing something which need high performance, so who care?

Haxalot - February 29, 2016, 8:18 am

(Feb 29, 2016 04:55 AM)Sakuto Wrote: [\[CITE\]](#) Strange that nobody suggested the trim function. It's perfect here. `trim($yourVariable, "");`

I avoided suggesting `trim()` because we aren't sure where the double quotes may occur in the string. We only know that OP wants to remove the first and last double quote of a string. `trim()` would therefore screw up the following string:

Code "Quoting Abc: "def""

Kedox - February 29, 2016, 11:03 am

This should be the answer you are looking for:

<http://stackoverflow.com/a/9735519>

DJSanderrr - February 29, 2016, 1:47 pm

(Feb 29, 2016 09:18 AM)Haxalot Wrote: [\[CITE\]](#) I avoided suggesting `trim()` because we aren't sure where the double quotes may occur in the string. We only know that OP wants to remove the first and last double quote of a string. `trim()` would therefore screw up the following string:

Code "Quoting Abc: "def""

No I actually only want to trim if it occurs at the beginning or the end of the string.

I apologize if I was unclear at the beginning.

Anyny0 - March 1, 2016, 9:40 am

(Feb 29, 2016 02:47 PM)DJSanderrr Wrote: [\[CITE\]](#) No I actually only want to trim if it occurs at the beginning or the end of the string.

I apologize if I was unclear at the beginning.

Which is why trim isn't exactly appropriate, as the following example wouldn't be the expected result:

"Quoting Abc: "def""

becomes

Quoting Abc: "def

^Source: Haxalot

ASDA - March 1, 2016, 11:41 am

Just out of curiosity, why do you need speed in an interpreted language?

DJSanderrr - March 2, 2016, 1:07 am

(Mar 1, 2016 12:41 PM)ASDA Wrote: [\[CITE\]](#) Just out of curiosity, why do you need speed in an interpreted language?

Custom import script for mysql databases which contain about 200k or more entries.

(Mar 1, 2016 10:40 AM)Anyny0 Wrote: [\[CITE\]](#) Which is why trim isn't exactly appropriate, as the following example wouldn't be the expected result:

"Quoting Abc: "def""

becomes

Quoting Abc: "def

^Source: Haxalot

That's correct, but currently I do not meet this kind of texts. But when I will I'll keep this in mind and might change it.

ASDA - March 2, 2016, 9:10 am

(Mar 2, 2016 02:07 AM)DJSanderrr Wrote: [\[CITE\]](#) Custom import script for mysql databases which contain about 200k or more entries.

That's correct, but currently I do not meet this kind of texts. But when I will I'll keep this in mind and might change it.

Would another language not work if you need raw speed?

DJSanderrr - March 2, 2016, 9:16 am

(Mar 2, 2016 10:10 AM)ASDA Wrote: [\[CITE\]](#) Would another language not work if you need raw speed?

I primarily script PHP because of it's simplicity, and because I'm the best at PHP.

What language would you recommend for me ? (also ubuntu-cli languages).

It has to be able to use mysql.

Sakuto - March 2, 2016, 11:34 am

Any language is able to use mysql database, but for scripting I would recommend python, perl, bash...

Terebic - March 2, 2016, 5:20 pm

I came accros this problem before when I was writing a lexer for a project of mine. I found my own way of doing it, take it for what it's worth, may be helpfull:

It would take input like:

PHP Code

```
<?php
someFuncton('lala', 'meow', true, 3);
```

And give the output:

PHP Code

```
<?php
array(
    0=>'someFunction',
    1=> array('lala', 'meow', true, 3)
);
```

Here's the function itself:

PHP Code

```
<?php
class foo
{
    protected function lexer( $string ) {
        $jar = explode('(', $string);
        $function = $jar[0];$args = trim(str_replace($function, "", $string), '()').', NULL';
        $arg_list = array();$in_quotes = array(false, "");$buff = "";
        foreach(str_split($args) as $char) {
            if($in_quotes[0]) {
                if($char !== $in_quotes[1]) {
                    $buff .= $char;
                } else {
                    $arg_list[] = $buff;
                    $buff = "";
                    $in_quotes = array(false, "");
                }
            } elseif($char === '"' || $char === "'") {
                $in_quotes[0] = true;
                $in_quotes[1] = $char;
            } elseif($char !== ',') {
                if($char !== ' ') {
                    $buff .= $char;
                } else {
                    if($buff !== "" && str_replace(' ', "", $buff) !== "") {
                        if($buff == 'true') {
                            $arg_list[] = true;
                        } elseif($buff == 'false') {
                            $arg_list[] = false;
                        } elseif($buff == 'null') {
                            $arg_list[] = null;
                        } else {
                            $arg_list[] = (int) $buff;
                        }
                    }
                    $buff = "";
                }
            }
        }
    }
}
```

```
        }  
    }  
}  
if($buff !== "") {  
    $arg_list[] = $buff;  
}  
return array($function, $arg_list);  
}  
}
```

A bit lengthy and hackish, but may help you a bit.