

Re: regexp again :( i am about to explode.....

## Re: regexp again :( i am about to explode.....

---

*Source:* <http://coding.derkeiler.com/Archive/Tcl/comp.lang.tcl/2007-02/msg00471.html>

---

- *From:* "Larry W. Virden" <[lvirden@xxxxxxxxxx](mailto:lvirden@xxxxxxxxxx)>
  - *Date:* 12 Feb 2007 04:52:55 -0800
- 

On Feb 9, 8:58 am, "SuNnY" <[esu...@xxxxxxxxxx](mailto:esu...@xxxxxxxxxx)> wrote:

Why strange things happen to me ???

In at least 80% of the cases, "strange things" happen because the developer (which includes me :blush:) tries to do something not really thinking through the consequences, or, in some cases, without knowing all the gritty details, and thus when tcl does exactly what it is asked to do, the results may *\_seem\_* strange, but in reality are absolutely correct. So, the better thing for you and I to do is to clearly figure out what it is we want to do, then to read closely the various commands so we use the right type of commands in the right conditions, so that we know how to ask tcl to do the right thing.

first of all let me tell what i am working on ...i am trying to match occurrence of "port #(any hex number)" for e.g "port #45af " or "port #abce" in a file.

Now, that's a good start. If I understand you correctly, you have a text file. In that text file are a series of lines that has various text. Some of these lines will have, among the other words, the words "port" and the string "#hex number" separated by one or more blanks, right? And you wish to skip lines without the port and #hex, and when you find the lines with those two "words", you wish to get the two words out. Is that accurate?

this is what i have in my code..

```
set Pattern {\mport #[0x0-9a-fA-F]}
```

Why \m? When I type:

Re: regexp again :( i am about to explode.....

Re: regexp again :( i am about to explode.....

```
% set a "\mABC"  
mABC
```

Are you trying to say "I only want to match lines where the first 4 characters are "port"? So, for instance, if the line says:  
here is another one – port #123a  
then you don't want to match this line?

Also, you have specified, for the value after the #, a single character which can be a 0 or an x or a 0–9 or an a–f or an A–F. No need to specific 0 twice. I suspect that what you were trying to say was that you wanted to match

```
port #0
```

```
or
```

```
port #1
```

```
:
```

and so on and that you also wanted to match

```
port #0xabcd
```

If you really only are worrying, in Pattern, about the `_first_` character after #, and you don't want to match things like:

```
port # abcd
```

for instance, then just use

```
set Pattern {^port #[:xdigit:]}
```

```
puts $Pattern  
foreach Str [fileutil::grep $Pattern [fileutil::find $RootDir  
{string match *.inform}]] { ;#<===== this works fine, as Str has  
all those lines where $Pattern occurs.  
puts "$Str" ;#<===== Example of the problem  
port #fa56 lets see it through <====content of single line file
```

```
set patMat [lsearch –exact –inline –all –regexp $Str {[port[ ]+[\#]  
[a–fA–F0–9]+}]] ;#<===== works absolutely the way i want it to  
when tried at wish console
```

If you are determined to use `lsearch`, then first do a split on `$Str`, so that the whole thing becomes a true Tcl list.

doesnt match anything when tried to run the script. `$patMat` its empty.

Note that the pattern you give `lsearch` is different than you used to select.

For instance, in the `lsearch`, the pattern indicates that port can

Re: regexp again :( i am about to explode.....

Re: regexp again :( i am about to explode.....

occur anywhere, that it can be followed by 1 \_or more\_ spaces (however you only grepped one space) and lsearch would expect that the port and port number are in the same list entry, but they won't be. I don't know how it ever worked in a wish console, since the code intrinsically would fail for this code... unless you somehow typed as a string to match against something like "{port #abcd} and other stuff" another peculiarity. Your grep pattern said that the line you say is an example hit had the word "port" at the beginning of the line – but your grep pattern indicated there would be a "m " before port. The next example below has a "m " in it...

```
***its definitely matching "port#af56" when i write set  
patMat[lsearch.....-regexp $Str {[port#[a-fA-F0-9]+}]}
```

In this case, you don't have ANY spaces after port – while before you list that there must be at least one space.

I tried on a file hving just this line "Example of the problem port #fa56 lets see it thorough"

This is an excellent example of why it is important for us to first write up as detailed requirements as we can do, before we start coding. That way, if we have examples of what kind of input we have to match, then we have examples in front of us to code against.

At Wish console:

```
(bin) 43 % regexp {port[ ]+[\#][a-fA-F0-9]+} "Example of the  
problem port #fa56 lets see it thorough" match
```

```
1
```

```
(bin) 44 % puts $match  
port #fa56
```

Its windows platform, Tcl v 8.4.14, Komodo IDE, version 4.0.1, build 274919, platform win32-x86.

regexp != lsearch. I recommend, strongly, sticking to regexp in your case.

.

Re: regexp again :( i am about to explode.....