

http::geturl parse 1 line at a time after keyword?

Source: <http://coding.derkeiler.com/Archive/Tcl/comp.lang.tcl/2006-07/msg00013.html>

- *From:* Rosc <rosc@xxxxxxxxxxxxx>
 - *Date:* Sat, 01 Jul 2006 09:13:18 GMT
-

Hi All

I'm trying to port a script that parses wunderground.com's html output. The original script used tcl's "socket" code. I'm attempting to translate it to use the tcl http package.

My question is this: How to step through several lines after finding a keyword.

For example, the original code used:

```
if {[regexp {Local Time:} $wzout]} {
for {set t 0} {$t <= 4} {incr t} {
set wzout [gets $wzsock]
regexp {<b>(.*?)</b>} $wzout match localtime
regexp {(.*?) on} $localdate match localtime
if {![regexp on $localdate]} {
set localtime $localdate
}
}
}
```

The original above basically grabs "Local Time:" and then pulls in up to 5 more lines, one line at a time, looking for "data" to get the right data.

This is what the html looks like, for the example:

```
<td class="taR" style="white-space: nowrap;">Local Time:</td>
<td id="full" style="white-space: nowrap;">
<b>2:38 AM EDT</b>
```

I would prefer to not use the htmlParse package, since this will be running in an eggdrop bot, and don't want to require other packages if it can be avoided.

Since I'm going to be using the http package, the entire html source will be in

http::geturl parse 1 line at a time after keyword?

a var. So my question is, how could I step 1 line at a time, up to a specific number of lines, after finding the keyword?

I've seen some examples of using multi-line regex's, and that may work. There are some instances on wunderground where the format changes, however. For example, in the 'Lunar phases' section:

The usual format:

```
<td style="padding: 3px; border-bottom: 1px solid #CCC;">Moon</td>
<td style="border-bottom: 1px solid #CCC;">11:19 AM EDT</td>
<td style="border-bottom: 1px solid #CCC;">No Moon Set</td>
</tr>
```

Unusual format:

```
<td style="padding: 3px; border-bottom: 1px solid #CCC;">Moon</td>
<td style="border-bottom: 1px solid #CCC;">
10:14 AM AKDT (6/30)
</td>
<td style="border-bottom: 1px solid #CCC;">
2:56 AM AKDT (6/30)
</td>
</tr>
```

This is the original code to handle the above:

```
# moonrise/set
set i 0
set wzout [gets $wzsock]
# Find the keyword "Moon"
while {[regexp {>Moon</td>} $wzout] == 0} && ($i < 40) {
incr i
set wzout [gets $wzsock]
}
set mrset "0"
set msset "0"
set msfp "1"
# step 1 line at a time til we get the data:
for {set t 0} {$t <= 4} {incr t} {
set wzout [gets $wzsock]
if {$mrset == "0"} {
set mrset [regexp {>(.*?)</td>} $wzout match moonrise]
if {$mrset == 0} {
#hope this line doesn't word-wrap:
set mrset [regexp {[[[:space:]]{2}([[:digit:]]{1,2}):[[:digit:]]{2} .*)} $wzout match moonrise]
# word-wrap?
}
}
if {($msset == "0") && ($mrset == 1)} {
if {$msfp == 1} {
```

http::geturl parse 1 line at a time after keyword?

http::geturl parse 1 line at a time after keyword?

```
set wzout [gets $wzsock]; incr t;incr msfp
}
set msset [regexp {>(.*?)</td>} $wzout match moonset]
if {$msset == 0} {
#same here, word-wrap possible:
set msset [regexp {[[[:space:]]{2}([[[:digit:]]{1,2}:[[[:digit:]]{2} .*)]} $wzout match moonset]
# word-wrap?
}
}
}
```

Hoping that someone can show me an example to step 1 line at a time through the http::geturl variable/data, after finding the keywords, similar to how the original worked.

I've also thought about using the 'foreach line \$html' type code, but I'm trying to minimize how many times the \$html has to be parsed, to make this faster. Plus I would think that using "foreach line" will give me too many bad matches. I need to be able to limit the matches to within a specific number of lines after finding the keyword.

Thanks in advance for any ideas/clues. :)

-C.L.

.