

Re: Assistance needed with script.....

Source: <http://coding.derkeiler.com/Archive/Perl/perl.beginners/2005-09/msg00404.html>

- *From:* security.department@xxxxxxxxx (John Doe)
 - *Date:* Mon, 19 Sep 2005 10:02:16 +0200
-

Bryan Jones am Sonntag, 18. September 2005 23.46:

> That worked well – Thanks. Could you assist me into understanding what
> exactly that has done.
>
> if (/^\\s+Device Capacity/) { <-- I understand this, this is a normal
> regex search.
> local \$/ = '}; <-- I understand this line, it
> makes the \$INPUT_RECORD_SEPARATOR local to this block
> \$device{ \$symmSerial }{ symmvol }{ \$symmvol }{ devicecapacity } = <--
> I understand this, this is the hoh structure
> { map split(\\s*:\\s*/), map split(\\s*\\n\\s*/), <> =~ <--
> Here is where I start to get lost, I have a basic understanding of the map
> function.
>
> From this point on is where I need a better
> understanding of what is going on.
> \\s*{\\s+(.)}/s };
> }
>
> If anyone can assist into my knowlegde of what is going on here, I would
> greatly appreciate it.

JWK's one-statement-solution is really impressive and shows that he is one of the big gurus on this list :-)

The first purpose of my answer is to learn. I `_try_` to explain; didn't test it and think that Cylinders and Tracks are, finally, also in the structure. If not, I did not understand JWK's solution.

Below the "map EXPR,LIST" syntax of map is used (see `perldoc -f map`).

The comments below should be read from top to bottom, as the expression is evaluated in this direction.

```
if ( /^\\s+Device Capacity/ ) {  
local $/ = '};  
$device{ $symmSerial }{ symmvol }{ $symmvol }{ devicecapacity } =
```

Re: Assistance needed with script.....

```
{
# the pairs are placed into {} and thus resulting in a hashref

map split( /\s*:\s*/ ),
# every line is split into the label and the number,
# forming a list of pairs (label, number).
# The "map EXPR,LIST" here consists of:
# EXPR: above line
# LIST: The list of real lines gotten so far

map split( /\s*\n\s*/ ),
# the "line" in $1, now implicitly referred as $_,
# is split into several "real" lines,
# resulting in a list of lines.

<>=~/\s*{\s+(.)}/s
# stores everything between the '{}' into $1,
# out of the the "line" consisting of the part '{...}'
# (the "line" ends with '}' due to $/ eq '}').
# The /s modifier must be used since the "line" consists
# of several "real" lines.
# '<>' stands for a "line".
# The match expression returns $1, a list with one element

};
}
```

joe

[below the original question and JWK's answer]

```
> "John W. Krahn" <krahnj@xxxxxxxxxx> wrote in message
> news:432CD4D3.10505@xxxxxxxxxxxxxx
>
> Bryan Jones wrote:
> > Hello all,
>
> > Hello,
>
> > I am somewhat new to perl. I have tried to read everything that I can.
> > I have a somewhat simple script, so might think. I have attached the
> > script and the file that I am reading to gather the data. I am trying to
> > parse this file to gather information and from here I will import the
> > data into a
> > spreadsheet. I need assistance on this one area with in the script.
> >
> > I have a few areas in the data file that has information that looks like
> > the
> > following:
> >
```

Re: Assistance needed with script.....

Re: Assistance needed with script.....

```
>> Device Capacity
>> {
>> Cylinders : 16
>> Tracks : 240
>> 512-byte Blocks : 15360
>> MegaBytes : 8
>> KiloBytes : 7680
>> }
>>
>> What I am trying to do is to gather the data in between the { } so it
>> would
>> look like such
>> SymmSerial =>
>> symmvol =>
>> $symmvol =>
>> {DeviceCapacity} =>
>> {512-byteBlocks} => 15360,
>> {MegaBytes} => 8,
>> {KiloBytes} => 7680,
>>
>> Above, there is only one for each symmvol. I have multiple items like
>> this
>> I would like to parse from the data file, such as Front Director Paths,
>> Device Capacity(from above) and Back End Disk Director Information.
>
> Here is one way to do it:
>
>
> if ( /^s+Device Capacity/ ) {
> local $/ = ' ';
> $device{ $symmSerial }{ symmvol }{ $symmvol }{ devicecapacity } =
> { map split( /s*:\s*/ ), map split( /s*\n\s*/ ), <> =~
> /s*{\s+(.)}/s };
> }
.
```

• **References:**

- ◆ **Assistance needed with script.....**
 ◇ From: Bryan Jones
- ◆ **Re: Assistance needed with script.....**
 ◇ From: John W. Krahn
- ◆ **Re: Assistance needed with script.....**
 ◇ From: Bryan Jones

- Prev by Date: **Re: displaying a message using perl code**
- Next by Date: **regd. cgi -apache**
- Previous by thread: **Re: Assistance needed with script.....**
- Next by thread: **generate date sequences**
- Index(es):

Re: Assistance needed with script.....

- ◆ Date
- ◆ Thread