

Re: regular expression help

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework/2008-02/msg00084.html>

- *From:* Jesse Houwing <jesse.houwing@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 5 Feb 2008 23:35:44 +0000 (UTC)
-

Hello Jeremy,

If you perform a match, you will still get 5 matches though on text such as
 1,4,"32,760.00"
 You will get "1", "", "4", "", "32.760.00"
 I don't understand why it returns 2 empty strings.. any ideas?

I don't know why it does. It has something to do with the internal rules of the .Net Regex engine. But all you have to do is check for a 0 Length on each match.

Kevin Spencer
Chicken Salad Surgeon
Microsoft MVP

Basically because if you remove everything that is optional in the regex below you end up with an empty regex:

```
\s*(?:(<word>"[^"]*"|(<!">(<word>[^,"]*)(<!">))\s*
```

\s* is optional

(?:(<word>"[^"]*"|(<!">(<word>[^,"]*)(<!">)) is optional because one of the parts of the alteration (namely (<!">(<word>[^,"]*)(<!">) is optional)

\s* is optional

So the regex engine will try to match on every character in the string:

1 stop, first match is found
 , comma doesn't match, but the nothingness in front of it does.
 4 stop, third match is found
 , comma doesn't match, but the nothingness in front of it does.
 "32,760.00" and the last match is found.

Re: regular expression help

You can easily demonstrate this by replacing instead of matching. Replace the matches with something like \$ and all the places where a match are found are visible:

using the above regex and \$\$ as replacement pattern you will get the following result:
\$\$,\$\$, \$

So as you can see that it matches a second value just before each ", ".

The issue can be solved in two ways:

1) make sure there are no optional parts:

```
\s*(?:(?<word>"[^\"]*"|(?<!"|)(?<word>[^\s,]+)(?<!"|))\s*
```

2) match the content of what you want to match even more closely within the pattern of beginning of the line, comma separated values, end of the line:

```
(?<=^|,)\s*(?<word>"[^\"]*"|(?<word>[^\s,]*?))\s*(?=,|$)
```

This last regex also removes any whitespace around the non quoted values and removes the quotes from quoted values.

I hope this explains it for you. If you have any more questions don't hesitate to ask.

—

Jesse Houwing
jesse.houwing at sogeti.nl

Code:

```
System.Text.RegularExpressions.MatchCollection pMatches =  
System.Text.RegularExpressions.Regex.Matches(strText, strRegex,  
System.Text.RegularExpressions.RegexOptions.ExplicitCapture);  
foreach (System.Text.RegularExpressions.Match pMatch in pMatches)  
{  
System.Text.RegularExpressions.Group pGroup = pMatch.Groups["word"];  
string strValue = pGroup.Value;  
}  
"Kevin Spencer" <unclechutney@localhost> wrote in message  
news:O5MCTjzYIHA.1532@xxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Use the following:

```
\s*(?:(?<word>"[^\"]*"|(?<!"|)(?<word>[^\s,]*)(?<!"|))\s*
```

Rather than splitting, it captures all of the elements without the commas (and spaces) between them. The way it works is this:

Re: regular expression help

It uses a non-capturing group to indicate that either of the two choices may be preceded and followed by 0 or more spaces. This eliminates preceding and following spaces from the groups.

It will capture one of two options:

A quote followed by any sequence of characters that is not a quote, followed by a quote.
Any sequence of characters that is NOT preceded by a quote and does not contain either quotes or commas, and is NOT followed by a quote.
The group "word" will give you all the matches that you want.

-- HTH,

Kevin Spencer
Chicken Salad Surgeon
Microsoft MVP

"Jeremy" <nospam@xxxxxxxxxxx> wrote in message
news:eErTaugYIHA.5208@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

I created a regular expression to parse a line in a csv file;

```
(\"(?<word>[^\"]+|\\\")*\"|(?<word>[^\,]*))
```

It is capable of taking a line such as
field1,field2,field
3,123.12,\"1,234.56\" and matching each value between the commas into the word group, so I get
field1
field2
field 3
123.12
1,234.56
My problem is that if I perform a split, or match on a string like
1,1,\"123.345\" I will get 6 matches back instead of 3.

```
const string strDelimiter =  
"(\\\"(?<word>[^\"]+|\\\")*\"|(?<word>[^\,]*))";
```

Re: regular expression help

```
string strText = "1,1,\"12,212.43\"";
string[] strParts =
System.Text.RegularExpressions.Regex.Split(strText
,
strDelimiter,System.Text.RegularExpressions.RegexOptions.Compiled
|
System.Text.RegularExpressions.RegexOptions.ExplicitCapture);
System.Text.RegularExpressions.MatchCollection
pMatches =
System.Text.RegularExpressions.Regex.Matches(strText
,
strDelimiter,System.Text.RegularExpressions.RegexOptions.ExplicitCa
pture);
Split returns 13 values, as shown below, and
Matches returns 6
items.
```

How can I just extract the 3 items?

```
strParts {Dimensions:[13]} string[]
[0] "" string
[1] "1" string
[2] "" string
[3] "" string
[4] "," string
[5] "1" string
[6] "" string
[7] "" string
[8] "," string
[9] "30,478.50" string
[10] "" string
[11] "" string
[12] "" string
```

—
Jesse Houwing
jesse.houwing at sogeti.nl