RegularExpressions

Regular Expressions in JMeter

JMeter includes the pattern matching software [http://jakarta.apache.org/oro/ Apache Jakarta ORO].

There is some documentation for this on the Jakarta web-site.

There is also documentation on an older incarnation of the product at [http://www.savarese.org/oro/docs/OROMatcher/index.html OROMatcher User's guide], which might prove useful.

Overview

The pattern matching is very similar to the pattern matching in Perl. A full installation of Perl will include plenty of documentation on regular expressions - look for perlrequick, perlretut, perlre, perlreref. O'Reilly sell a book called "Mastering Regular Expressions" by Jeffrey Friedl which will tell you all you need to know (and a lot more) about regular expressions.

There are also a couple of sample chapters available on their web-site covering REs in Java and .NET, and the Java chapter has a [http://www.oreilly.com /catalog/regex2/chapter/ch08.pdf section on ORO (PDF)] - worth a look.

It is worth stressing the difference between "contains" and "matches", as used on the [http://jakarta.apache.org/jmeter/usermanual/component_reference. html#Response_Assertion Response Assertion] test element:

- "contains" means that the regular expression matched at least *some* part of the target, so 'alphabet' "contains" 'ph.b.' because the regular expression matches the substring 'phabe'.
- "matches" means that the regular expression matched the *whole* target. So 'alphabet' is "matched" by 'al.*t'. In this case, it is equivalent to wrapping the regular expression in ^ and $, viz '^al.*t$'. However, this is not always the case. For example, the regular expression 'alp|.lp.*' is "contained" in 'alphabet', but **does not match** 'alphabet'.

Why? Because when the pattern matcher finds the sequence 'alp' in 'alphabet', it stops trying any other combinations - and 'alp' is not the same as 'alphabet', as it does not include 'habet'.

Note: unlike Perl, there is no need to (i.e. do not) enclose the regular expression in //. So how does one use the Perl modifiers ismx etc if there is no trailing /? The solution is to use Perl5 extended regular expressions, i.e. /abc/i becomes (?i)abc

Links to regex resources

http://www.regular-expressions.info/tutorial.html

http://tlc.perlarchive.com/articles/perl/pm0001_perlretut.shtml

For an extremely useful Regex tester, see http://weitz.de/regex-coach/

http://www.visibone.com/regular-expressions/ - quick reference

Examples

Suppose you want to match the following portion of a web-page: `name="file" value="readme.txt"` and you want to extract `readme.txt`.

A suitable requlat expression would be:

name="file" value="`(.+?)`"

The special characters above are:

- `(` and `)` - these enclose the portion of the match string to be returned
- `.` - match any character. + - one or more times. ? - don't be greedy, i.e. stop when first match succeeds

Note: without the ?, the `.+` would continue past the first " until it found the last possible " - probably not what was intended.