Positive/Negative Lookahead/Lookbehind in Vim

Nov 1, 2016

vim handles lookaround expressions a little bit differently than PCRE (which languages like Ruby and Perl use). As a quick recap, positive lookaround expressions help solve the issue of making sure that your match always follows/is followed by another expression. Conversely, negative lookaround makes sure that your match **isn't** surrounded by the given expression.

For example, say I have a document: `hello foo, welcome to foobar` and I only wanted to match the second "foo" which is followed by "bar". I could use positive lookahead (i.e. "match me 'foo' where the next expression is 'bar', but don't include 'bar' in the match"). We might naively think we can use character classes or captures (e.g. `/(foo)bar/`) but these have edge cases, particularly when we deal with negative lookaround expressions.

In the preceding example we might use positive lookahead to solve this issue. In PCRE this looks like:

```
/foo(?=bar)/
```

vim however doesn't implement PCRE. Instead vim's lookaround expressions affect the previous capture group, so first we'd need to capture "bar" then apply the lookahead modifier to it:

```
/foo\(bar\)\@=/
```

All of the looakaround expressions work in a similar way: capture, then apply the appropriate lookaround modifier.

(Note that vim treats parenthesis literals differently than PCRE. Where PCRE always treats them as special characters unless escaped, vim always treats them as literals unless escaped.)

For reference:

- Positive lookahead: `\@=`

- Negative lookahead: `\@!`
- Positive lookbehind: `\@<=`
- Negative lookbehind: `\@<!`

jbodah.github.io

jbodah.github.io
jb3689@yahoo.com

jbodah

Some guy's thoughts on Elixir, Ruby, and other things