

### Subpatterns

Subpatterns are used to group alternatives, to match a previously-matched pattern again, and to extract match text using **Demo**.

Subpatterns are delimited by parentheses, which can be nested. Turning part of a pattern into a subpattern localizes a set of alternatives. For example, this pattern (which includes two vertical bars signifying alternation):

```
cat(aract|erpillar|)
```

matches one of the words "cat", "cataract", or "caterpillar". Without the parentheses, it would match "cataract", "erpillar" or the empty string.

### Named Subpatterns

Specifying subpatterns by number is simple, but it can be very hard to keep track of the numbers in complicated regular expressions. Furthermore, if an expression is modified, the numbers may change. To help with this difficulty, PCRE supports the naming of subpatterns, something that Perl does not provide. The Python syntax (*?P<name>...*) is used. For example:

```
My (?P<catdog>cat|dog) is cooler than your (?P=catdog)
```

Here catdog is the name of the first and only subpattern. *?P<catdog>* names the subpattern and *(?P=catdog)* matches the previous match for that subpattern.

Names consist of alphanumeric characters and underscores, and must be unique within a pattern.

Named capturing parentheses are still allocated numbers as well as names. This has the same effect as the previous example:

```
My (?P<catdog>cat|dog) is cooler than your \1
```

### Repetition

Repetition is specified by quantifiers:

?	0 or 1 quantifier
	Example: <code>[abc] ?</code> - Matches 0 or 1 occurrences of a or b or c
*	0 or more quantifier
	Example: <code>[abc] *</code> - Matches 0 or more occurrences of a or b or c
+	1 or more quantifier
	Example: <code>[abc] +</code> - Matches 1 or more occurrences of a or b or c
{n}	n times quantifier
	Example: <code>[abc] { 3 }</code> - Matches 3 occurrences of a or b or c
{n,m}	n to m times quantifier
	Example: <code>[abc] { 3, 5 }</code> - Matches 3 to 5 occurrences of a or b or c

Quantifiers can follow any of the following items:

- A literal data character
- The `.` metacharacter
- The `\C` escape sequence
- An escape such as `\d` that matches a single character
- A character class