RegexpCommon

Non serve essere utenti Perl di vecchissima data per sapere quale sia la critica rivolta più spesso al nostro linguaggio: "Perl è illeggibile", "Sembra il rumore di una linea telefonica", "Hai il gatto che di notte ti cammina sulla tastiera". E via discorrendo.

Ritengo che una delle ragioni di questo luogo comune sia nel supporto integrato delle espressioni regolari: sono comode, possono risultare efficienti e, ad un livello sufficientemente alto di maestria del programmatore, sono indistinguibili dalla magia. Il problema è che non solo sono difficili da leggere (ma qualche accortezza la si può osservare: http://www.perl.it/documenti/faq/view.html?pag=6&faq=1), ma sono anche piuttosto difficili da scrivere.

Un modo per risolvere entrambi i problemi è usare il modulo Regexp::Common. Si tratta di una collezione di espressioni regolari di uso comune, catalogate e raggruppate per contesto di applicazione.

Ad esempio, all'inizio di questa pagina della nostra FAQ sono enumerate una serie di espressioni regolari per il riconoscimento di numeri. Eccone una:

```
if (/^-?(?:\d+(?:\.\d*)?|\.\d+)$/) { print "e` un numero decimale\n" }
```

che e` abbastanza complessa da sembrare scritta dal nostro gatto magico. Riscriviamo lo stesso pezzo di codice con Regexp::Common:

```
use Regexp::Common;
if ( /$RE{num}{real}/ ) { print "e` un numero decimale\n" }
```

Immediatamente leggibile, e chiaro anche a chi non ha dimestichezza con le espressioni regolari. E' cosa che può tornare utile, in un progetto che coinvolge più persone con competenze diverse.

Altrettanto utile, e per lo stesso motivo, la possibilità fornita da Regexp::Common di incapsulare le proprie espressioni regolari nell'hash %RE gestito dal modulo, adoperando la funzione pattern().

```
use strict;
use warnings;

use Data::Dumper;
use Regexp::Common 'pattern';

pattern name   => [ qw/tel cell -operator/ ] ,
        create => sub {
            my $self  = shift;
            my $flags = shift;
            my %prefix = (
                Wind      => [ 328 ] ,
                Vodaphone => [ 348, 349 ] ,
                Telecom   => [ 338, 335 ] ,
                Fake      => [ 555 ]
            );

            my $pref = '';
            if ( defined $flags->{ -operator } ) {
                $pref = join "|", @{$prefix{$flags->{ -operator }}};
            }
            else {
                $pref = join "|", map { @$_ } values %prefix;
            }
            return '^'
```

```
                    . "($pref)"
                    . '\d+$';
            }
            ;

    use Test::More qw/ no_plan /;

    ok( "3492323298" =~ $RE{tel}{cell}{-operator => 'Vodaphone'} );
    ok( "5553429844" =~ $RE{tel}{cell}{-operator => 'Fake'}      );
    ok( "3383429844" =~ $RE{tel}{cell}                           );
```

[larsen]