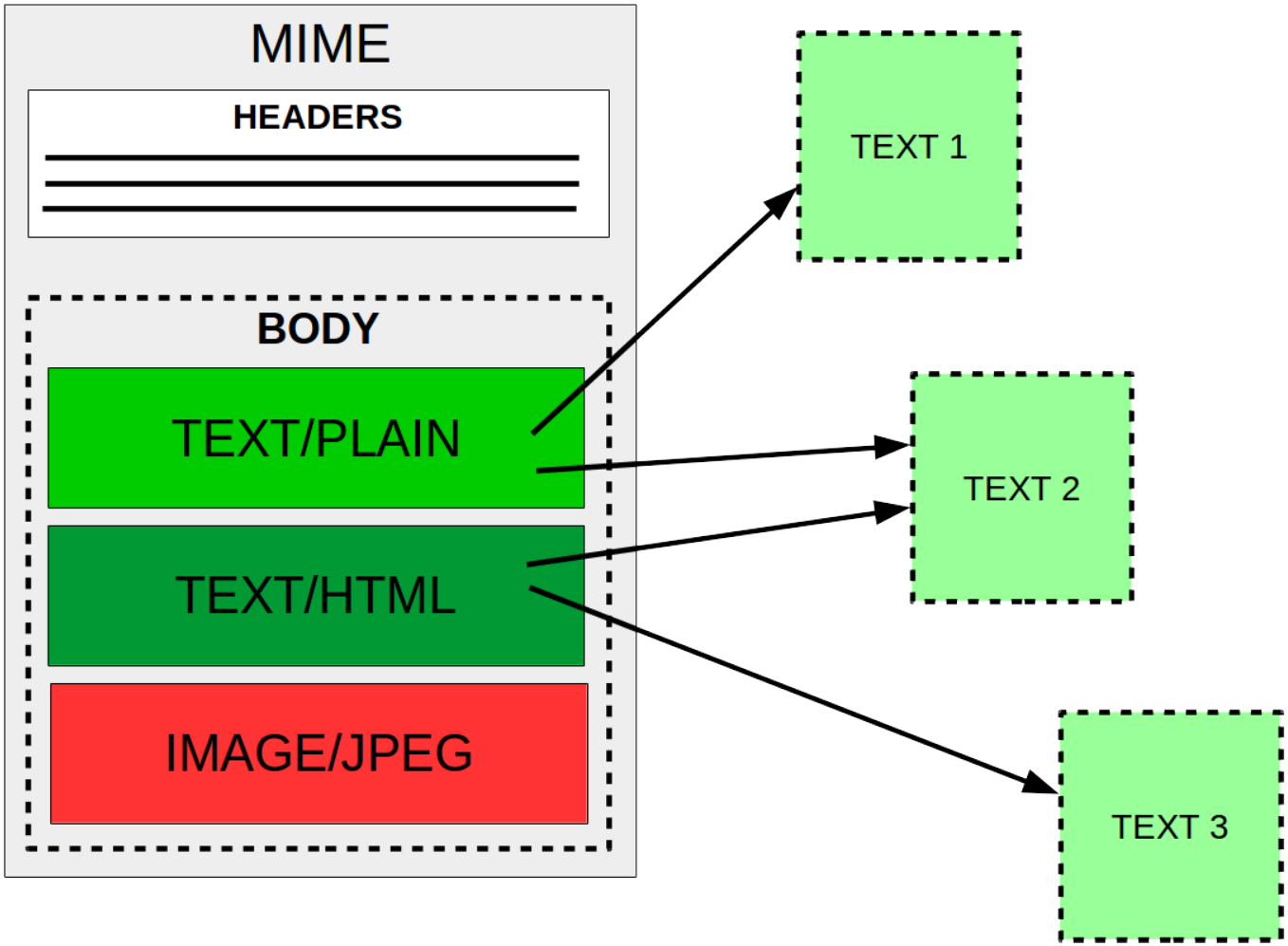
Writing Nice Regular Expressions

Dmitry Shiltsov

April 14, 2015

Me and regular expressions

Kaspersky Anti-Spam (2006 – 2014)



Readability vs performance vs coding time

- Simple regexps
- Optimized regexps
- Not using regexps

Regular expressions in Perl

```
1    m/PATTERN/;
2    s/PATTERN/REPLACEMENT/;
3    split /PATTERN/, EXPR;
4    qr/STRING/;
```

Matching phone numbers

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',

Matching phone numbers: regexp 1

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',

```
1 qr/\(04\) \d{3}-\d{4}\b/;
```

Matching phone numbers: regexp 1

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',

```
1 qr/\(04\) \d{3}-\d{4}\b/;
```

Matching phone numbers: regexp 2

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',

```
1 qr/(?:\(04\) |\+64-4-)\d{3}[ -]\d{4}\b/;
```

Matching phone numbers: regexp 3

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',

```
1 qr/(?:\(04\) |\+64-4-)(?:\d{3}[ -]\d{4}|\d{4}[ -]\d
      {3})\b/;
```

Matching phone numbers: regexp 4

Wellington region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',

```
1 qr/(?:\(04\)|\+?\b64[ -]4)[ -](?:\d{3}[ -]\d{4}|\d{4}[
      -]\d{3})\b/;
```

Matching phone numbers: regexp 4

Wellington *and Auckland* region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',
'09 123 4567',
'(+64 9) 123 4567',

```
1 qr/(?:\(04\)|\+?\b64[ -]4)[ -](?:\d{3}[ -]\d{4}|\d{4}[
      -]\d{3})\b/;
```

Matching phone numbers: regexp 5

Wellington and Auckland region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',
'09 123 4567',
'(+64 9) 123 4567',

```
1 qr/(?:\(?\b0[49]\)?|\+?\b64[ -][49])[ -](?:\d{3}[ -]\d
      {4}|\d{4}[ -]\d{3})\b/;
```

Matching phone numbers: regexp 6

Wellington and Auckland region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',
'09 123 4567',
'(+64 9) 123 4567',

```
1 qr/\(?(?:0[49]|\+?\b64[ -][49])\)?[ -](?:\d{3}[ -]\d
      {4}|\d{4}[ -]\d{3})\b/;
```

Matching phone numbers: regexp 7

Wellington and Auckland region land line:

'(04) 529-9999',
'(04) 239-9999',
'(04) 569-9999',
'(04) 479-9999',
'(04) 389-9999',
'+64-4-123 4567',
'64 4 1234 567',
'(04) 1234 401',
'09 123 4567',
'(+64 9) 123 4567',

```
1 qr/\(?(?:\+?\b64[ -]?|\b0)[49]\)?[ -]?(?:\d{3}[ -]?\d
      {4}|\d{4}[ -]?\d{3})\b/;
```

Matching phone numbers: free formatting and comments

```
qr/\(?(?:\+?\b64[ -]?|\b0)[49]\)?[ -]?(?:\d{3}[ -]?\d
    {4}|\d{4}[ -]?\d{3})\b/;
```

```
qr/
    \(?
        (?:\+?\b64[ -]?|\b0) # prefix +64 or 0
        [49]                 # 4 - Well, 9 - Auck
    \)?[ -]?
    (?:
        \d{3}[ -]?\d{4}  # numbers like xxx-xxxx
        |
        \d{4}[ -]?\d{3}  # numbers like xxxx-xxx
    )
    \b/x;
```

```
qr/(Abc(?# Another way to put comments )Xyz)/;
```

Matching phone numbers: with variables

```
qr/\(?(?:\+?\b64[ -]?|\b0)[49]\)?[ -]?(?:\d{3}[ -]?\d
    {4}|\d{4}[ -]?\d{3})\b/;
```

```
my $sep = qr/[ -]+/;
my $land_prefix = qr/[49]/;    # 4 - Well, 9 - Auck
my $prefix = qr/
     \+?\b64 $sep $land_prefix # intenational prefix
     |
     \b0$land_prefix/x;        # local prefix
my $main_number = qr/
     \d{3} $sep \d{4}          # numbers like xxx-xxxx
     |
     \d{4} $sep \d{3}/x;       # numbers like xxxx-xxx

my $regexp= qr/\(? $prefix \)? $sep $main_number \b/x;
```

Extracting substrings

```
1  my $date = 'Tue Mar 10 18:57:40 NZDT 2015';
2
3  my $date_pattern = qr /^
4       ([A-Z][a-z][a-z])\s # weekday
5       ([A-Z][a-z][a-z])\s # month
6       (\d{1,2})\s         # day
7       (\d\d:\d\d:\d\d)\s  # time
8       (\S+)\s             # tz
9       (\d{4})/x;          # year
10
11 if ($date =~ $date_pattern) {
12      say "$3 $2 $6";
13 }
```

```
1  10 Mar 2015
```

Extracting substrings: named captures

```
my $date = 'Tue Mar 10 18:57:40 NZDT 2015';

my $date_pattern = qr /^
     (?<weekday>[A-Z][a-z][a-z])\s
     (?<month>[A-Z][a-z][a-z])\s
     (?<day>\d{1,2})\s
     (?<time>\d\d:\d\d:\d\d)\s
     (?<tz>\S+)\s
     (?<year>\d{4})/x;

if ($date =~ $date_pattern) {
     say "$+{day} $+{month} $+{year}";
}
```

```
10 Mar 2015
```

Some regexp modifiers

```
1  /PATTERN/msixgo
```

- m treat string as multiple lines.
- s treat string as single line
- i case-insensitive
- x free formating
- o compile once
- g iteration

```
1  /Abc (?i-sm)case insensitive here(?-i) Xyz /ms;
2  /Abc (?i-sm:case insensitive here) Xyz /ms;
```

Email: From matches To

```
my $headers =
'Received: from somewhere ([123.123.123.123])
        by someone-else with lots of details...
From: Canadian Pharmacy <doctor@superpharm.info>
Reply-To: Fake Name <email@server.com>
To: Another Name <doctor@superpharm.info>
Subject: Enlarge your pencil!
Content-Type: text/plain';
```

```
my $email = qr/<[\w.-]+@[\w\.-]+\.[a-z]{2,4}>/;

my $from_in_to = qr/
    \nFrom: .* ($email)\r?\n   # capturing from-address
    (?:.*\r?\n)*               # any number of lines
    To: .*\1                   # To matches From
/x;
```

Email: From matches To (with /s and /m)

```
my $headers =
'Received: from somewhere ([123.123.123.123])
        by someone-else with lots of details...
From: Canadian Pharmacy <doctor@superpharm.info>
Reply-To: Fake Name <email@server.com>
To: Another Name <doctor@superpharm.info>
Subject: Enlarge your pencil!
Content-Type: text/plain';
```

```
my $email = qr/<[\w.-]+@[\w\.-]+\.[a-z]{2,4}>/;

my $from_in_to = qr/
    ^From: \N* ($email)$ # capturing from-address
    .*
    ^To: \N* \1$/xms;    # To matches From
```

Dealing with slashes

```
/PATTERN/
m/PATTERN/
m!PATTERN!
```

```
s/PATTERN/REPLACEMENT/
s{PATTERN}[REPLACEMENT]
```

Examples:

```
m/^\/home\/dmitry\/Desktop\/.*pdf$/
m\/home/dmitry/Desktop/.*pdf$\
```

Beware of a special case!

```
m?PATTERN?
```

Literal search (quotemeta)

```
my $code = '... \d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}...'
    ;
$ip =~ m/\\d\{1,3\}\\\.\\d\{1,3\}\\\.\\d\{1,3\}\\\.\\d
    \{1,3\}/
$ip =~ m/\Q\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\E/
```

```
my $perl_line = 'say "${+{day} ${+{month} ${+{year}";';
my $pattern = qr/^\s*\Q$say_line\E$/m;
```