

Perl 5.10

# Versions of Perl (perlhist)

**5.000** 17 October 1994

**5.6.0** 22 March 2000

**5.8.0** 18 July 2002

**5.8.8** 31 January 2006

**5.10** 18 December 2008

# Versions of Perl (perlhist)

**5.000**    **17 October 1994**

**5.6.0**    **22 March 2000**

**5.8.0**    **18 July 2002**

**5.8.8**    **31 January 2006**

**5.10**    **18 December 2008**

**6.0**    **?**

# Versions of Perl (perlhist)

**5.000** 17 October 1994

**5.6.0** 22 March 2000

**5.8.0** 18 July 2002

**5.8.8** 31 January 2006

**5.10** 18 December 2008

**6.0** Christmas

# Versions of Perl (perlhist)

**5.000** 17 October 1994

**5.6.0** 22 March 2000

**5.8.0** 18 July 2002

**5.8.8** 31 January 2006

**5.10** 18 December 2008

**6.0** 2000

use feature

```
use feature qw(  
    say  
    switch  
    state  
);
```

```
use feature qw(  
    say  
    switch  
    state  
);
```

```
use feature "5.10";
```

```
use feature qw(  
    say  
    switch  
    state  
);
```

```
use feature "5.10";  
use 5.10.0;
```

```
use feature qw(  
    say  
    switch  
    state  
);
```

```
use feature "5.10";  
use v5.10.0;
```

```
#!/per15.10/bin/perl
```

```
use feature "***";
```

```
#!/per15.10/bin/perl
```

```
use feature "say";
```

```
say "Perl 6?";
```

```
#!/per15.10/bin/perl
```

```
use feature "say";
```

```
say "Perl 6?";
```

```
no feature "say";
```

```
> perl5.10 -e \  
"use feature qw(say); say $$;"
```

```
> perl5.10 -e \  
"use feature qw(say); say $$;"
```

```
> perl5.10 -E "say $$;"
```

//

defined-or

```
my $a;
```

```
my $b = $a // 2;
```

```
say $b; 2
```

```
my $c = 0;
```

```
my $d = $c // 3;
```

```
say $d; 0
```

```
my $e = 0;
```

```
my $f = $e || 4;
```

```
say $f; 4
```

my \$\_;

```
for (1..5) {  
    my $_ = '*';  
    print;  
}
```

\*\*\*\*\*

\$ :: : \_

```
for (1..5) {  
  my $_ = '*';  
  print $::$_;  
}
```

12345

our \$ \_ ;

```
for (1..5) {  
  our $_ = '*';  
  print $::_;  
}
```

\*\*\*\*\*

```
use strict 'refs';
```

```
my $x = '***';
```

```
print 1 if defined $$x;
```

```
use strict 'refs';  
my $var = '***';  
print defined $$var  
      ? 'yes' : 'no';
```

```
> perl5.8.8 test.pl
```

```
no
```

```
use strict 'refs';  
my $var = '***';  
print defined $$var  
      ? 'yes' : 'no';
```

```
> perl5.10 test.pl
```

```
Can't use string ("xxx") as a  
SCALAR ref while "strict refs"  
in use at test.pl line 3.
```

```
use feature 'switch';
```

```
use feature qw(switch say);
```

```
my $tag = 'hrpw2008';
```

```
given ($tag) {
```

```
    when ('hrpw2008') {
```

```
        say 'Yes';
```

```
    }
```

```
}
```

```
use feature qw(switch say);
```

```
my $tag = 'hrpw2008';
```

```
given ($tag) {
```

```
    when ('hrpw2008') {
```

```
        say 'Yes';
```

```
    }
```

```
    default {say 'No';}
```

```
}
```

when (123)

when (\$value)

when (undef)

when ([2001..2100])

when (/d+/)

when ( $\$_$  > 0)

when (int)

when (int  $\$_$ )

when (&test\_the\_value)

when (test\_the\_value( $\$_$ ))

```
given ( 'hrpw2008' ) {  
  when ( /\d+/ ) {  
    say 'digits';  
    continue;  
  }  
  when ( /perl/i ) {  
    say 'Perl';  
  }  
}
```

when (\$what)

==

when (\$\_ ~~ \$what)

$\$left \sim\sim \$right$

$= =$

$\$right \sim\sim \$left$

```
use feature 'state';
```

```
sub counter{  
  state $value = 0;  
  $value++;  
  say $value;  
}
```

```
counter();      1  
counter();      2  
counter();      3
```

# Regular Expressions

# Named saving parens

```
my $date = 'Thu 15 April 2008';
```

```
$date =~ /
```

```
    (          \w+ )    \s+
```

```
    (          \d+ )    \s+
```

```
    (          \w+ )    \s+
```

```
    (          \d{4})
```

```
/x;
```

```
say $1;
```

Thu

```
say $4;
```

2008

# Named saving parens

```
my $date = 'Thu 15 April 2008';
```

```
$date =~ /
```

```
    (?<wday>    \w+ )    \s+
```

```
    (?<day>     \d+ )    \s+
```

```
    (?<month>   \w+ )    \s+
```

```
    (?<year>    \d{4})
```

```
/x;
```

```
say $+{wday};
```

Thu

```
say $+{year};
```

2008

# Named saving parens

```
my $date = 'Thu 15 April 2008';
```

```
$date =~ s/  
    (?<year>\d{4})  
    /  
    ${year} + 1  
    /xe;
```

```
say $date;           Thu 15 April 2009
```

```
my $code = 'my $value = 100; say $value;';
```

```
$code =~ s/
```

```
my \s*  
(?<variable> \#[a-z]+) \s*  
= \s*  
(?<value> [^;]+ ) \s*  
; \s*
```

```
(?<other_code>.*?)
```

```
(\k<variable>)
```

```
/$+{other_code}$+{value}/x;
```

```
say $code;
```

```
say 100;
```

```
my $leap_years = '1992 1996 2004 2008';
```

```
$leap_years =~ m/
```

```
(
```

```
    ?<year>    1    \d{3}
```

```
)
```

```
\s*
```

```
(
```

```
    ?<year>    2    \d{3}
```

```
)
```

```
/x;
```

```
say $_ for @{$leap_years};
```

1996

2004

```
my $leap_years = '1992 1996 2004 2008';
```

```
$leap_years =~ m/
```

```
(
```

```
    ?<year>    1    \d{3}
```

```
)
```

```
/gx;
```

```
say $_ for @{${year}};
```

```
1992
```

```
my $leap_years = '1992 1996 2004 2008';
```

```
$leap_years =~ m/
```

```
(
```

```
    ?<year>    1    \d{3}    \s*
```

```
)+
```

```
/gx;
```

```
say $_ for @{$leap_years};
```

```
1996
```

```
use feature 'say';
```

```
my $expr = '1 + (2 + (3 + (4 + 5) + 6))';
```

```
$expr =~ s/
```

```
  \(
```

```
    (
```

```
      [^()]+
```

```
    )
```

```
  |
```

```
    (?1)
```

```
  \)
```

```
  /say $1;/xge;
```

# Possessive quantifiers

?+

\*+

++

{min, max}+

/

"

(?:

[^"\\]+

|

\\.

)\*

"

/x

( ? | . . . )

```
my $re = qr/  
    (\d{4})(\d\d)(\d\d)  
    |  
    (\w+),\s*(\d{4})  
/x;
```

```
'20080415' =~ $re;  
say "$1 . $2 . $3";
```

```
'April, 2008' =~ $re;  
say "$4 . $5";
```

```
my $re = qr/  
  (? | (\d{4})(\d\d)(\d\d))  
  |  
  (\w+), \s*(\d{4}))  
/x;
```

```
'20080415' =~ $re;  
say "$1 . $2 . $3";
```

```
'April, 2008' =~ $re;  
say "$1 . $2";
```

$\backslash g\{N\}$

$\backslash gN$

$\setminus g\{-N\}$

`\k<named>`

`==`

`\g{named}`

\K

\v

\h

\V

\H

\R

\R

(?>

\x0D\x0A?

|

[

\x0A-\x0C

\x85

\x{2028}

\x{2029}

]

)

# Smart matching



$\$a \approx \approx \$b$

$= =$

$\$b \approx \approx \$a$

```
my $b;
```

```
$b ~~ undef
```

```
!defined $b
```

```
my $c = 'abc';
```

```
$c =~ 'abc'
```

```
$c eq 'abc'
```

```
my $c = 'abc';
```

```
$c =~ /b/
```

```
$c =~ /b/
```

my @a = (1..3);

my @b = (1..3);

@a == @b

1 == 1 && 2 == 2 && 3 == 3

my @a = (1..3);

my @b = (1..3);

my @c = (3..5);

@a ~~ @c

1 == 3 && 2 == 4 && 3 == 5

```
my @d = (123, 'abc');
```

```
my @e = (qr/\d/, qr/\w/);
```

```
@d ~ @e
```

```
123 ~ /\d/ &&
```

```
'abc' ~ /\w/
```

```
my @f = ('a'..'f');
```

```
@f =~ 'd'
```

```
grep {$_ eq 'd'} @f
```

```
my @g = (1..10);
```

```
@g =~ 7
```

```
grep {$_ == 7} @g
```

```
my @g = (1..10);
```

```
@g =~ 7.0
```

```
grep {$_ == 7.0} @g
```

```
my @g = (1..10);
```

```
@g =~ '7.0'
```

```
grep {$_ eq '7.0'} @g
```

```
my @g = (1..10);
```

```
@g =~ /^\d$/
```

```
grep {$_ =~ /^\d$/} @g
```

3.14  $\approx\approx$  '3.14'

3.14  $==$  '3.14'

3.14 ~ ~ '3.14%'

3.14 == '3.14%'

```
sub subA {return 2}  
sub subB {return 2}
```

subA ~~~ subB

subA() == subB()

```
sub subA {return 2}
my $subA1_ref = \&subA;
my $subA2_ref = \&subA;
```

```
$subA1_ref ~~ $subA2_ref
```

```
$subA1_ref == $subA2_ref
```

```
sub subA {return 2}  
my $subA_ref = \&subA;
```

```
$a ~~ $subA_ref
```

```
$subA_ref->($a)
```

```
sub subA {return 2}  
my $subA_ref = \&subA;
```

```
-1 ~~ $subA_ref
```

```
$subA_ref->(-1)
```

```
my %h = (a => 'alpha',  
         b => 'beta');
```

```
%h =~ 'a'
```

```
exists $h{'a'}
```

```
my %h = (a => 'alpha',  
         b => 'beta');
```

```
my @f = ('a'..'f');
```

```
%h ~~ @f
```

```
grep {$_} @h{@f}
```

```
my %h = (a => 'alpha',  
         b => 'beta');
```

```
%h =~ /[A-F]/i
```

```
grep {/[A-F]/i} keys %h
```

```
my %h = (a => 'alpha',  
         b => 'beta');
```

```
my %hh = (b => 1, a => 2);
```

%h ~ ~ %hh

[sort keys %h] ~ ~

[sort keys %hh]

# Elements of Perl 6 in Perl 5.10

# Elements of Perl 6 in Perl 5.10 and differences

say

```
my $x = 'HRPW2008';
```

```
say $x;
```

```
my $x = 'HRPW2008';  
say $x;
```

HRPW2008

```
my $x = 'HRPW2008';  
say $x;
```

HRPW2008

```
my $x = 'HRPW2008';  
say ($x);
```

```
my $x = 'HRPW2008';  
say ($x);
```

**HRPW2008**

```
my $x = 'HRPW2008';  
say ($x);
```

**HRPW2008**

```
my $x = 'HRPW2008';  
say($x);
```

**HRPW2008**

```
my $x = 'HRPW2008';  
$x.say;
```

**HRPW2008**

```
my $x = 'HRPW2008';  
$x.say;
```

```
my $x = 'HRPW2008';  
$x.say;
```

**String concatenation!**

```
my $x = 'HRPW2008';  
$x.say();
```

**HRPW2008**

\$

—

```
for (1..3) {  
    say;  
}
```

```
for (1..3) {  
    say;  
}
```

1

2

3

```
for (1..3) {  
  say;  
}
```

```
\n
```

```
\n
```

```
\n
```

```
for (1..3) {  
  say $_;  
}
```

1

2

3

```
for (1..3) {  
    $_.say;  
}
```

1

2

3

```
for (1..3) {  
  .say;  
}
```

1

2

3

switch

5.10, 6

```
my $str = "YAPC::Asia";  
given ($str) {  
    when (/Asia/) {  
        say "Asia"  
    }  
}
```

5.10, 6

```
my $str = "YAPC::Asia";  
given ($str) {  
  when (/Asia/) {  
    say "Asia"  
  }  
}
```

```
my $str = "YAPC::Asia";  
given ($str) {  
    when (/Asia/) {  
        say "Asia"  
    }  
}
```

```
my $str = "YAPC::Asia";  
given $str {  
  when /Asia/ {  
    say "Asia"  
  }  
}
```

```
my $str = "YAPC::Asia";  
given $str {  
    say "Asia" when /Asia/  
}
```

statae

```
sub f {  
    state $c;  
    say ++$c;  
}
```

```
sub f {  
    state $c;  
    say ++$c;  
}  
f(); f(); f();
```

# 5.10

```
sub f {  
    state $c;  
    say ++$c;  
}  
  
f(); f(); f();  
1 2 3
```

```
sub f {  
    state $c;  
    say ++$c;  
}  
  
f(); f(); f();  
1 2 3
```

```
sub f {  
    state $c = 0;  
    say ++$c;  
}  
f(); f(); f();
```

# 5.10

```
sub f {  
    state $c = 0;  
    say ++$c;  
}
```

```
f(); f(); f();
```

1 2 3

# pugs

```
sub f {  
    state $c = 0;  
    say ++$c;  
}
```

```
f(); f(); f();
```

**1 1 1**

BERGAMO



Croatian Perl Workshop,  
Zagreb, 2008

\_\_\_END\_\_\_

Andrew Shitov

mail@andy.sh | <http://andy.sh>