

*Understand Your Data and
Be More Productive*

3rd Edition
For Perl, PHP, Java,
.Net, Ruby, and More!



Mastering

Regular Expressions

O'REILLY®

Jeffrey E.F. Friedl

Index

- ❖ xxii
- \(-\) 137
- \<-\> 21, 25, 50, 133-134, 150
 - egrep* 15
 - Emacs 101
 - mimicking in Perl 341-342
- \+ 141
- \\ 190, 380, 444
- \? 141
- '\+' history 87
- \0 117-118
- \1 138, 300, 303
 - (see also backreferences)
 - Perl 41
- \A 112, 129-130
 - (see also enhanced line-anchor mode)
 - optimization 246
- \a 115-116
- \B 134
- \b 65, 115-116, 134
 - (see also: word boundaries; backspace)
 - backspace and word boundary 44, 46
 - Java 368
 - Perl 286
 - PHP 442
- \b\B 240
- \C 120
 - PHP 442
- \D 49, 120
- \d 49, 120
 - Perl 288
 - PHP 442
- \E 290
 - (see also literal-text mode)
 - Java 368, 395, 403
- \e 79, 115-116
- \f 115-116
 - introduced 44
- \G 130-133, 212, 315-316, 362, 447
 - (see also pos)
 - advanced example 132, 399
 - .NET 408
 - optimization 246
- \kname (see named capture)
- \l 290
- \L\E 290
 - inhibiting 292
- \n 49, 115-116
 - introduced 44
 - machine-dependency 115
- \N{LATIN SMALL LETTER SHARP S} 290
- \N{name} 290
 - (see also pragma)
 - inhibiting 292
- \p{^} 288
- \p{...} 121, 288
 - (see also Unicode, properties)
 - Java 368-369, 402-403
 - Perl 125
- \p{All} 125
 - Perl 288
- \p{all} 369
- \p{Any} 125, 442
 - Perl 288
- \p{Arrows} 124

- \p{Assigned} 125-126
 - Perl 288
- \p{Basic_Latin} 124
- \p{Box_Drawing} 124
- \p{C} 122
 - Java 369
- \p{Cc} 123
- \p{Cf} 123
- \p{Cherokee} 122
- \p{Close_Punctuation} 123
- \p{Cn} 123, 125-126, 369, 408
 - Java 369
- \p{Co} 123
- \p{Connector_Punctuation} 123
- \p{Control} 123
- \p{Currency} 124
- \p{Currency_Symbol} 123
- \p{Cyrillic} 122, 124
- \p{Dash_Punctuation} 123
- \p{Decimal_Digit_Number} 123
- \p{Dingbats} 124
- \p{Enclosing_Mark} 123
- \p{Final_Punctuation} 123
- \p{Format} 123
- \p{Gujarati} 122
- \p{Han} 122
- \p{Hangul_Jamo} 124
- \p{Hebrew} 122, 124
- \p{Hiragana} 122
- \p{InArrows} 124
- \p{InBasic_Latin} 124
- \p{InBox_Drawing} 124
- \p{InCurrency} 124
- \p{InCyrillic} 124
- \p{InDingbats} 124
- \p{InHangul_Jamo} 124
- \p{InHebrew} 124
- \p{Inherited} 122
- \p{Initial_Punctuation} 123
- \p{InKatakana} 124
- \p{InTamil} 124
- \p{InTibetan} 124
- \p{IsCherokee} 122
- \p{IsCommon} 122
- \p{IsCyrillic} 122
- \p{IsGujarati} 122
- \p{IsHan} 122
- \p{IsHebrew} 122
- \p{IsHiragana} 122
- \p{IsKatakana} 122
- \p{IsLatin} 122
- \p{IsThai} 122
- \p{IsTibetan} 124
- \p{javaJavaIdentifierStart} 369
- \p{Katakana} 122, 124
- \pL PHP 442
- \p{L} 121-122, 133, 368, 395
- \p{L&} 122-123, 125, 442
 - Java 369
 - Perl 288
- \p{Latin} 122
- \p{Letter} 122, 288
- \p{Letter_Number} 123
- \p{Line_Separator} 123
- \p{LL} 123, 406
- \p{Lm} 123, 406
- \p{Lo} 123, 406
- \p{Lowercase_Letter} 123
- \p{Lt} 123, 406
- \p{Lu} 123, 406
- \p{M} 120, 122
- \p{Mark} 122
- \p{Math_Symbol} 123
- \p{Mc} 123
- \p{Me} 123
- \p{Mn} 123
- \p{Modifier_Letter} 123
- \p{Modifier_Symbol} 123
- \p{N} 122, 395
- \pN PHP 442
- \p{Nd} 123, 368, 406
- \p{NL} 123
- \p{No} 123
- \p{Non_Spacing_Mark} 123
- \p{Number} 122
- \p{Open_Punctuation} 123
- \p{Other} 122
- \p{Other_Letter} 123
- \p{Other_Number} 123
- \p{Other_Punctuation} 123
- \p{Other_Symbol} 123
- \p{P} 122
- \p{Paragraph_Separator} 123
- \p{Pc} 123, 406
- \p{Pd} 123
- \p{Pe} 123
- \p{Pf} 123
 - Java 369
- \p{Pi} 123
 - Java 369
- \p{Po} 123
- \p{Private_Use} 123
- \p{Ps} 123
- \p{Punctuation} 122
- \p{S} 122
- \p{Sc} 123-124

- `\p{Separator}` 122
- `\p{Sk}` 123
- `\p{Sm}` 123
- `\p{So}` 123
- `\p{Space_Separator}` 123
- `\p{Spacing_Combining_Mark}` 123
- `\p{Symbol}` 122
- `\p{Tamil}` 124
- `\p{Thai}` 122
- `\p{Tibetan}` 124
- `\p{Titlecase_Letter}` 123
- `\p{Unassigned}` 123, 125
 - Perl 288
- `\p{Uppercase_Letter}` 123
- `\p{Z}` 121-122, 368, 407
- `\pZ` PHP 442
- `\p{z1}` 123
- `\p{Zp}` 123
- `\p{Zs}` 123
- `\Q` Java 368, 395, 403
- `\Q \E` 290
 - inhibiting 292
- `\r` 49, 115-116
 - machine-dependency 115
- `\s` 49, 56, 121
- `\s` 49, 121
 - Emacs 128
 - introduction 47
 - Perl 288
 - PHP 442
- `\t` 49, 115-116
 - introduced 44
- `\U` 117
- `\u` 117, 290, 406
- `\U \E` 290
 - inhibiting 292
- `\v` 115-116, 364
- `\V` 364
- `\w` 49, 65, 120
 - Emacs 129
 - Java 368
 - many different interpretations 93
 - Perl 288
 - PHP 442
- `\W` 49, 121
- `\x` 117, 406
 - Perl 286
- `\x` 108, 120
- `\z` 112, 129-130
 - (see also enhanced line-anchor mode)
 - Java 370
 - optimization 246
- `\z` 112, 129-130, 316, 447
 - (see also enhanced line-anchor mode)
 - optimization 246
 - PHP 442
- `//` 322
- `/c` 131-132, 315
- `/e` 319-321
- `/g` 61, 132, 307, 311-312, 315, 319
 - (see also `\G`)
 - introduced 51
 - with regex object 354
- `/i` 135
 - (see also: case-insensitive mode; mode modifier)
 - introduced 47
 - with `study` 359
- `/m` 135
 - (see also: enhanced line-anchor mode; mode modifier)
- `/o` 352-353
 - with regex object 354
- `/osmosis` 293
- `/s` 135
 - (see also: dot-matches-all mode; mode modifier)
- `/x` 135, 288
 - (see also: comments and free-spacing mode; mode modifier)
 - history 90
 - introduced 72
- `-Dr` 363
- `-i` as `-y` 86
- `-Mre=debug` (see use `re 'debug'`)
- `-y` old *grep* 86
- `<>` 54
 - and `$_` 79
- `
` 481
- `!~` 309
- `#` (see comments)
- `$&` 299-300
 - checking for 358
 - mimicking 302, 357
 - naughty 356
 - .NET 424
 - OK for debugging 331
 - pre-match copy 355
- `$+` 300-301, 345
 - example 202
 - .NET 202, 424
- `$'` 300
 - checking for 358
 - mimicking 357
 - naughty 356

- \$` (cont'd)
 - .NET 424
 - OK for debugging 331
 - pre-match copy 355
- \$* 362
- \$/ 35, 78
 - Perl 35
- \$' 300
 - checking for 358
 - mimicking 357
 - naughty 356
 - .NET 424
 - OK for debugging 331
 - pre-match copy 355
- \$\$.NET 424
- \$ 112-113, 130, 447
 - (see also enhanced line-anchor mode)
 - escaping 77
 - Java 370
 - optimization 246
 - Perl interpolation 289
 - PHP 442
- \$_ 79, 308, 311, 314, 318, 322, 353-354, 359
 - .NET 424
- +[0] (see @+)
- \$0 300
 - Java 380
 - PHP 459
- [0] (see @-)
- #{0} 459
- \$1 137-138, 300, 303
 - introduced 41
 - Java 380
 - .NET 424
 - in other languages 138
 - pre-match copy 355
- \$all_matches 455
- \$ARGV 79
- \$HostnameRegex 76, 137, 303, 351
- \$HttpRequest 303, 305, 345, 351
- \$LevelN 330, 343
- \$matches 450
- ^N 300-301, 344-346
- #{name} 409
- #{name~} 424
- \$NestedStuffRegex 339, 346
- ^R 302, 327
- ^W 297
- % Perl interpolation 289
- (?:...) (see non-capturing parentheses)
- (...) (see parentheses)
- (?!) 241, 333, 335, 340-341
- (?#...) 99, 136, 420
- (?1)
 - Java 402
 - PCRE 476
 - PHP 476
- (?1) PHP 482
- (?i) (see: case-insensitive mode; mode modifier)
- (?i:...) (see mode-modified span)
- (?-i) 446
- (?i) 446
- (?if then | else) (see conditional)
- (?m) (see: enhanced line-anchor mode; mode modifier)
- (?m:...) (see mode-modified span)
- (?n) 408
- (?<name>...) (see named capture)
- (?'name'...) (see named capture)
- (?P<...>) 451-452, 457
- (?P=name...) (see named capture)
- (?P<name>...) (see named capture)
- (?R) 475
 - PCRE 475
 - PHP 475
- (?s) (see: dot-matches-all mode; mode modifier)
- (?s:...) (see mode-modified span)
- (?x:...) (see mode-modified span)
- (?x) (see: comments and free-spacing mode; mode modifier)
- ++ (see possessive quantifiers)
- * (see star)
- + (see plus)
- ++ 483
 - (see also possessive quantifiers)
- ".*" (see double-quoted string example)
- .*
 - introduced 55
 - mechanics of matching 152
 - optimization 246
 - warning about 56
- .NET xvii, 405-438
 - \$+ 202
 - after-match data 138
 - benchmarking 237
 - character-class subtraction 406
 - code example 219
 - flavor overview 92
 - JIT 410
 - line anchors 130
 - literal-text mode 136
 - MISL 410
 - object model 417

- .NET (cont'd)
 - \p{...} 125
 - regex approach 96-97
 - regex flavor 407
 - search and replace 414, 423-424
 - URL example 204
 - version covered 405
 - word boundaries 134
 - (see also VB.NET)
- =~ 308-309, 318
 - introduced 38
- ? (see question mark)
- ?...? 308
- ?+ (see possessive quantifiers)
- @"... " 103
- @- 300, 302, 339
- @+ 300, 302, 314
- @ Perl interpolation 289
- [...=] 128
- [:...:] 127
- [:<:] 91
- [... .] 128
- ^ 112-113, 130
 - (see also enhanced line-anchor mode)
 - Java 370
 - optimization 246
- ^Subject: **example** 94, 151-152, 154, 242-243, 245, 289
 - Java 95
 - .NET 96
 - Perl 55
 - Perl debugger 361
 - PHP 97
 - Python 97
- {*min,max*} 20, 141
- | (see alternation)
- \$+ .NET 202

- \0 117-118
- \$0 300
 - Java 380
 - PHP 459
- (?1)
 - Java 402
 - PCRE 476
 - PHP 476
- \1 138, 300, 303
 - (see also backreferences)
 - Perl 41
- \$1 137-138, 300, 303
 - introduced 41
 - Java 380

- \$1 (cont'd)
 - .NET 424
 - in other languages 138
 - pre-match copy 355
- (?1) PHP 482
- 8859-1 encoding 29, 87, 106, 108, 123

- \a 115-116
- @ escaping 77
- \A 112, 129-130
 - (see also enhanced line-anchor mode)
 - optimization 246
- after-match data**
 - Java 138
 - .NET 138
 - PHP 138
- after-match variables**
 - Perl 299
 - pre-match copy 355
- Aho, Alfred 86, 180
- \p{a11} 369
- \p{A11} 125
 - Perl 288
- \$all_matches 455
 - collated 455
 - vs. \$matches 454
 - stacked 456
- alternation** 139-140
 - and backtracking 231
 - efficiency 222, 231
 - greedy 174-175
 - hand tweaking 261
 - introduced 13-14
 - order of 175-177, 223, 260, 482
 - for correctness 28, 189, 197
 - for efficiency 224
 - and parentheses 13
- analogy**
 - backtracking
 - bread crumbs 158-159
 - stacking dishes 159
 - ball rolling 262
 - building a car 31
 - charging batteries 179
 - engines 143-147
 - first come, first served 153
 - gas additive 150
 - learning regexes
 - Pascal 36
 - playing rummy 33
 - regex as a language 5, 27
 - regex as filename patterns 4

- analogy (cont'd)
 - regex-directed match (see NFA)
 - text-directed match (see DFA)
 - transmission 148-149, 228
 - transparencies (Perl's `local`) 298
- anchor (see also: word boundaries; enhanced line-anchor mode)
 - caret 129
 - dollar 129
 - end-of-line optimization 246
 - exposing 256
 - line 87, 112-113, 150
 - overview 129
- anchored(...) 362
- anchored *'string'* 362
- anchoring bounds 388
 - Java 388
- AND class set operations 125-126
- ANSI escape sequences 79
- `\p{Any}` 125, 442
 - Perl 288
- any character (see dot)
- `appendReplacement` method 380
- `appendTail` method 381
- `$ARGV` 79
- array context (see list context)
 - `\p{Arrows}` 124
- ASCII encoding 29, 106-107, 115, 123
- Asian character encoding 29
- `AssemblyName` 435
- `\p{Assigned}` 125-126
 - Perl 288
- asterisk (see star)
- atomic grouping (see also possessive quantifiers)
 - details 170-172
 - for efficiency 171-172, 259-260, 268-270
 - essence 170-171
 - introduced 139
- atomic grouping example 198, 201, 213, 271, 330, 340-341, 346
- AT&T Bell Labs 86
- author email xxiii
- auto-lookaheadification 410
- automatic possessification 251
- awk*
 - after-match data 138
 - `gensub` 182
 - history 87
 - search and replace 100
 - version covered 91
 - word boundaries 134
- `\b` 65, 115-116, 134
 - (see also: word boundaries; backspace)
 - backspace and word boundary 44, 46
 - Java 368
 - Perl 286
 - PHP 442
- ⌘ 111, 128, 290
- `\B` 134
- `\b\B` 240
- ` ..` 165-167
 - unrolling 270
- backreferences 118, 137
 - DFA 150, 182
 - introduced with *egrep* 20-22
 - vs. octal escape 412-413
 - remembering text 21
- backspace (see `\b`)
- backtracking 163-177
 - and alternation 231
 - avoiding 171-172
 - computing count 227
 - counting 222, 224
 - detecting excessive 249-250
 - efficiency 179-180
 - essence 168-169
 - exponential match 226
 - global view 228-232
 - introduction 157-163
 - LIFO 159
 - of lookaround 173-174
 - neverending match 226
 - non-match example 160-161
 - POSIX NFA example 229
 - saved states 159
 - simple example 160
 - simple lazy example 161
- balanced constructs 328-331, 340-341, 436, 475-478, 481
- balancing regex issues 186
- Barwise, J. 85
- base character 107, 120
- Basic Regular Expressions 87-88
- `\p{Basic_Latin}` 124
- `\b\B` 240
- benchmarking 232-239
 - comparative 249
 - compile caching 351
 - Java 235-236
 - for naughty variables 358
 - .NET 237, 410
 - with neverending match 227
 - Perl 360
 - PHP 234-235

- benchmarking (cont'd)
 - pre-match copy 356
 - Python 238-239
 - Ruby 238
 - Tcl 239
- Berkeley 86
- Better-Late-Than-Never 236
- 165-167
 - unrolling 270
- blocks 124, 288, 369, 402, 407
- BLTN 236
 - Java 236
- BOL 362
- \p{Box_Drawing} 124
- Boyer-Moore 245, 247
- brace (see interval)
- bracket expressions 127
- BRE 87-88
- bread-crumble analogy 158-159
-
 481
- bugs Java 365, 368-369, 387, 392, 399, 403
- Bulletin of Math. Biophysics* 85
- bump-along
 - avoiding 210
 - distrusting 215-218
 - introduction 148-149
 - optimization 255
 - in overall processing 242
- Byington, Ryan xxiv
- byte matching 120, 442, 452-453, 456

- ¢ 124
- \p{C} 122
 - Java 369
- \C 120
 - PHP 442
- C# (see also .NET)
 - strings 103
- /c 131-132, 315
- C comments
 - matching 272-276
 - unrolling 275-276
- caching 242-245
 - (see also regex objects)
 - benchmarking 351
 - compile 242-245
 - Emacs 244
 - integrated 243
 - Java 478
 - .NET 432
 - object-oriented 244
- caching (cont'd)
 - Perl 350-352
 - PHP 478
 - procedural 244
 - Tcl 244
 - unconditional 350
- callback PHP 463, 465
- Capture 437
- CaptureCollection 438
- capturing parentheses Java 377
- car analogy 83-84
- caret anchor introduced 8
- carriage return 109, 370
- case title 110
- case folding 290, 292
 - inhibiting 292
- case-insensitive mode 110
 - egrep* 14-15
 - /i 47
 - introduced 14-15
 - Ruby 110
 - with `study` 359
- cast 294-295
- categories (see Unicode, properties)
- \p{Cc} 123
- CDATA 483
- Celsius (see temperature conversion example)
- \p{Cf} 123
- chaining (of methods) 389
- character
 - base 120
 - classes xvii
 - (see also character class)
 - combining 107, 120
 - Inherited script 122
 - vs. combining characters 107
 - control 117
 - initial character discrimination 245-248, 252, 257-259, 332, 361
 - machine-dependent codes 115
 - multiple code points 108
 - as opposed to byte 29
 - separating with `split` 322
 - shorthands 115-116
- character class 118
 - vs. alternation 13
 - vs. dot 119
 - elimination optimization 248
 - introduced 9-10
 - and lazy quantifiers 167
 - mechanics of matching 149

- character class** (cont'd)
 - negated
 - must match character 11-12
 - and newline 119
 - Tcl 112
 - positive assertion 119
 - of POSIX bracket expression 127
 - range 9, 119
 - as separate language 10
 - set operations 125-127
 - subtraction 406
 - subtraction (set) 126
 - subtraction (simple) 125
- character equivalent** 128
- character-class subtraction** .NET 406
- CharBuffer** 373, 376, 387
- charnames pragma** 290
- CharSequence** 365, 373, 382, 397
- CheckNaughtiness** 358
- \p{Cherokee}** 122
- Chinese text processing** 29
- chr** 420
- chunk limit**
 - Java 396
 - Perl 323
 - PHP 466
- CJKV Information Processing** 29
- class** xvii
 - initial class discrimination 245-248, 252, 257-259, 332, 361
 - (see also character class)
- Click, Cliff** xxiv
- client VM** 236
- clock clicks** 239
- \p{Close_Punctuation}** 123
- closures** 339
- \p{Cn}** 123, 125-126, 369, 408
 - Java 369
- \p{Co}** 123
- code example**
 - Java 81, 209, 217, 235, 371, 375, 378-379, 381-384, 389
 - .NET 219
- code point**
 - beyond U+FFFF 109
 - introduced 107
 - multiple 108
 - unassigned in block 124
- coerce** 294-295
- cold VM** 236
- collated data** 455
- collating sequences** 128
- combining character** 107, 120
 - Inherited script 122
- commafying a number example** 64-65
 - introduced 59
 - without lookbehind 67
- COMMAND.COM** 7
- comments** 99, 136
 - Java 98
 - matching of C comments 272-276
 - matching of Pascal comments 265
 - .NET 420
 - XML 483
- comments and free-spacing mode** 111
- Communications of the ACM** 85
- comparison of engine types** (see NFA)
- Compilation failed** 474
- compile**
 - caching 242-245
 - once (/o) 352-353
 - on-demand 351
 - regex 410-411
- compile method** 372
- Compiled(.NET)** 237, 408, 410, 420, 427-428, 435
- Compilers—Principles, Techniques, and Tools** 180
- CompileToAssembly** 433, 435
- conditional** 140-141
 - with embedded regex 327, 335
 - mimicking with lookaround 140
 - .NET 409-410
- Config module** 290, 299
- conflicting metacharacters** 44-46
- \p{Connector_Punctuation}** 123
- Constable, Robert** 85
- context** (see also: list context; scalar context; match, context)
 - contorting
 - Perl 294
 - forcing 310
 - metacharacters 44-46
 - regex use 189
- continuation lines** 178, 186-187
 - unrolling 270-271
- contorting an expression** 294-295
- \p{Control}** 123
- control characters** 117
- Conway, Damian** 339
- cooking for HTML** 68, 414
- copy for \$&** (see pre-match copy)
- correctness** vs. efficiency 223-224
- counting quantifier** (see interval)
- www.cpan.org** 358

CR 109, 370
 create_function 463, 465
 CR/LF 370
 Cruise, Tom 51
 crummy analogy 158-159
 CSV parsing example
 Java 217, 401
 .NET 435
 Perl 213-219
 PHP 480
 unrolling 271
 VB.NET 219
 \p{Currency} 124
 currency
 € 123-124, 367, 406
 \p{Currency} 124
 \p{Currency_Symbol} 123
 \p{Sc} 123
 Unicode block 123-124
 \p{Currency_Symbol} 123
 current location Java 374, 383, 398, 400
 currentTimeMillis() 236
 \p{Cyrillic} 122, 124

 \D 49, 120
 \d 49, 120
 Perl 288
 PHP 442
 DARTH 197
 dash in character class 9
 \p{Dash_Punctuation} 123
 date_default_timezone_set 235
 DBIx::DWIW 258
 debugcolor 363
 debugging 361-363
 with embedded code 331-332
 regex objects 305-306
 run-time 362
 \p{Decimal_Digit_Number} 123
 default regex 308
 define-key 101
 delegate 423-424
 delimited text 196-198
 standard formula 196, 273
 delimiter
 with shell 7
 with substitution 319
 delimiters PHP 445, 448
 description Java 365
 Deterministic Finite Automaton (see DFA)
 Devel::FindAmperSand 358
 Devel::SawAmperSand 358

DFA
 acronym spelled out 156
 backreferences 150, 182
 boring 157
 compared with NFA 224, 227
 (see also NFA)
 efficiency 179
 implementation ease 183
 introduced 145, 155
 lazy evaluation 181
 longest-leftmost match 177-179
 testing for 146-147
 dialytika 108
 \p{Dingbats} 124
 directed alternation (see alternation,
 ordered)
 dish-stacking analogy 159
 dollar for Perl variable 37
 dollar anchor 129
 introduced 8
 dollar value example 24-25, 51-52,
 167-170, 175, 194-195
 DOS 7
 dot 119
 vs. character class 119
 introduced 11-12
 Java 370
 mechanics of matching 149
 Tcl 113
 dot 370
 dot modes Java 111, 370
 .NET xvii, 405-438
 \$+ 202
 after-match data 138
 benchmarking 237
 character-class subtraction 406
 code example 219
 flavor overview 92
 JIT 410
 line anchors 130
 literal-text mode 136
 MISL 410
 object model 417
 \p{...} 125
 regex approach 96-97
 regex flavor 407
 search and replace 414, 423-424
 URL example 204
 version covered 405
 word boundaries 134
 dot-matches-all mode 111-112

double-quoted string example

- allowing escaped quotes 196
- egrep* 24
- final regex 264
- makudonarudo 165, 169, 228-232, 264
- sobering example 222-228
- unrolled 262, 268

double-word finder example 81

- description 1
- egrep* 22
- Emacs 101
- Java 81
- Perl 35, 77-80

-Dr 363

dragon book 180

DWIW (DBIx) 258

dynamic regex 327-331

- sanitizing 337

dynamic scope 295-299

- vs. lexical scope 299

\E 290

(see also literal-text mode)

Java 368, 395, 403

\e 79, 115-116

/e 319-321

earliest match wins 148-149

EBCDIC 29

ECMAScript (.NET) 406, 408, 412-413, 421, 427

ed 85**efficiency** (see also optimization)

- and backtracking 179-180
- correctness 223-224
- Perl 347-363
- Perl-specific issues 347-363
- PHP 478-480
- regex objects 353-354
- unlimited lookbehind 134

egrep

- after-match data 138
- backreference support 150
- case-insensitive match 15
- doubled-word solution 22
- example use 14
- flavor overview 92
- flavor summary 32
- history 86-87
- introduced 6-8
- metacharacter discussion 8-22
- regex implementation 183
- version covered 91

egrep (cont'd)

- word boundaries 134

electric engine analogy 143-147**else** (see conditional)**Emacs**

- after-match data 138
- control characters 117
- flavor overview 92
- re-search-forward 101
- search 100
- strings as regexes 101
- syntax class 128
- version covered 91
- word boundaries 134

email of author xxiii**email address example** 70-73, 98

- Java 98
- .NET 99

embedded code

- local 336
- my 338-339
- regex construct 327, 331-335
- sanitizing 337

embedded string check optimization 247, 257

Embodiments of Mind 85

Empty 433

empty-element tag 481**encapsulation** (see regex objects)

\p{Enclosing_Mark} 123

encoding (see also Unicode)

- ASCII 29, 106-107, 115, 123
- introduced 29
- issues overview 105
- Latin-1 29, 87, 106, 108, 123
- UCS-2 107
- UCS-4 107
- UTF-16 107
- UTF-8 107, 442, 447

END block 358**end method** 377**end of line** (see anchor, dollar)**end of previous match** (see \G)**end of word** (see word boundaries)**end-of-string anchor optimization** 246**engine**

- analogy 143-147
- hybrid 182, 239, 243
- implementation ease 183
- introduced 27
- testing type 146-147
 - with neverending match 227
- type comparison 156-157, 180-183

English module 357
English vs. regex 275
enhanced line-anchor mode 112-113
 introduced 69
ERE 87-88
ereg suite 439
errata xxiii
Escape 432
escape
 introduced 22
 term defined 27
essence
 atomic grouping 170-171
 greediness, laziness, and backtrack-
 ing 168-169
 NFA (see backtracking)
eval 319
example
 atomic grouping 198, 201, 213, 271,
 330, 340-341, 346
 commafying a number 64-65
 introduced 59
 without lookbehind 67
 CSV parsing
 Java 217, 401
 .NET 435
 Perl 213-219
 PHP 480
 unrolling 271
 VB.NET 219
 dollar value 24-25, 51-52, 167-170, 175,
 194-195
 double-quoted string
 allowing escaped quotes 196
 egrep 24
 final regex 264
 makudonarudo 165, 169, 228-232,
 264
 sobering example 222-228
 unrolled 262, 268
 double-word finder 81
 description 1
 egrep 22
 Emacs 101
 Java 81
 Perl 35, 77-80
 email address 70-73, 98
 Java 98
 .NET 99
 filename 190-192, 444
 five modifiers 316
 floating-point number 194
 form letter 50-51

example (cont'd)
 gr[ea]y 9
 hostname 22, 73, 76, 98-99, 137-138,
 203, 260, 267-268, 304, 306,
 450-451
 egrep 25
 Java 209
 plucking from text 71-73, 206-208
 in URL 74-77
 validating 203-205
 VB.NET 204
 HREF 452
 HTML 443-444, 459, 461, 464, 481, 484
 conversion from text 67-77
 cooking 68, 414
 encoding 414
 <HR> 194
 link 201-203
 optional 140
 paired tags 165
 parsing 132, 315, 321, 399
 tag 9, 18-19, 26, 200-201, 326, 357
 URL 74-77, 203, 206-208, 303,
 450-451
 URL-encoding 320
 HTTP response 467
 image tags 397
 IP 5, 187-189, 267-268, 311, 314,
 348-349
 Jeffs 61-64
 lookahead 61-64
 mail processing 53-59
 makudonarudo 165, 169, 228-232, 264
 pathname 190-192
 population 59
 possessive quantifiers 198, 201
 postal code 209-212
 regex overloading 341-345
 stock pricing 51-52, 167-168
 with alternation 175
 with atomic grouping 170
 with possessive quantifier 169
 temperature conversion
 Java 382
 .NET 425
 Perl 37, 283
 PHP 444
 text-to-HTML 67-77
 this|that 133, 139, 243, 245-247, 252,
 255, 260-261
 unrolling the loop 270-271, 477

- example** (cont'd)
 - URL 74-77, 201-204, 208, 260, 303-304, 306, 320, 450-451
 - egrep* 25
 - Java 209
 - plucking 206-208
 - username 73, 76, 98
 - plucking from text 71-73
 - in URL 74-77
 - variable names 24
 - XML 481-484
 - ZIP code 209-212
- exception**
 - `IllegalArgumentException` 373, 380
 - `IllegalStateException` 376-377
 - `IndexOutOfBoundsException` 375-376, 380
 - `IOException` 81
 - `PatternSyntaxException` 371, 373
- `Explicit` (Option) 415
- `ExplicitCapture` (.NET) 408, 420, 427
- exponential match** 222-228, 330, 340
 - avoiding 264-266
 - discovery 226-228
 - explanation 226-228
 - non-determinism 264
 - short-circuiting 250
 - solving with atomic grouping 268
 - solving with possessive quantifiers 268
- expose literal text** 255
- expression**
 - context 294-295
 - contorting 294-295
- Extended Regular Expressions** 87-88

- `\f` 115-116
 - introduced 44
- Fahrenheit** (see temperature conversion example)
- failure**
 - atomic grouping 171-172
 - forcing 241, 333, 335, 340-341
- FF 109, 370
- file globs** 4
- file-check example** 2, 36
- filename**
 - patterns (*globs*) 4
 - prepending to line 79
- filename example** 190-192, 444
- Filo, David 397
- `\p{Final_Punctuation}` 123

- find method** 375
 - region 384
- `FindAmperсанд` 358
- Fite, Liz** 33
- five modifiers example** 316
- flags method** 394
- flavor**
 - Perl 286-293
 - superficial chart
 - general 92
 - Java 367
 - .NET 407
 - PCRE 441
 - Perl 285, 287
 - PHP 441
 - POSIX 88
 - term defined 27
- flex* version covered 91
- floating regex cache** (see regex objects)
- floating *'string'* 362
- floating-point number example** 194
- forcing failure** 241, 333, 335, 340-341
- foreach vs. while vs. if 320
- form letter example** 50-51
- `\p{Format}` 123
- freeflowing regex** 277-281
- Friedl, Alfred 176
- Friedl, brothers 33
- Friedl, Fumie v, xxiv
 - birthday 11-12
- Friedl, Jeffrey xxiii
- Friedl, Stephen xxiv, 458
- fully qualified name** 295
- functions** related to regexes in Perl 285

- `\G` 130-133, 212, 315-316, 362, 447
 - (see also `pos`)
 - advanced example 132, 399
 - .NET 408
 - optimization 246
- `/g` 61, 132, 307, 311-312, 315, 319
 - (see also `\G`)
 - introduced 51
 - with regex object 354
- garbage collection** Java benchmarking 236
- gas engine analogy** 143-147
- general categories** (see Unicode, properties)
- `gensub` 182
- George, Kit xxiv
- `GetGroupNames` 427-428

- GetGroupNumbers 427-428
- gettimeofday 234
- Gill, Stuart xxiv
- global match (see /g)
- global vs. private Perl variables 295
- globs filename 4
- GNU *awk*
 - after-match data 138
 - gensub 182
 - version covered 91
 - word boundaries 134
- GNU *egrep*
 - after-match data 138
 - backreference support 150
 - doubled-word solution 22
 - i bug 21
 - regex implementation 183
 - word boundaries 134
- GNU Emacs (see Emacs)
- GNU *grep*
 - shortest-leftmost match 182
 - version covered 91
- GNU *sed*
 - after-match data 138
 - version covered 91
 - word boundaries 134
- Gosling, James 89
- GPOS 362
- Greant, Zak xxiv
- greatest weakness Perl 286
- gr[ea]y example 9
- greedy (see also lazy)
 - alternation 174-175
 - and backtracking 162-177
 - deference to an overall match 153, 274
 - essence 159, 168-169
 - favors match 167-168
 - first come, first served 153
 - global vs. local 182
 - introduced 151
 - vs. lazy 169, 256-257
 - localizing 225-226
 - quantifier 141
 - swapping 447
 - too greedy 152
- green dragon 180
- grep Perl 324
- grep*
 - as an acronym 85
 - flavor overview 92
 - history 86
 - regex flavor 86
 - version covered 91
- grep* (cont'd)
 - y option 86
- group method 377
- Group object (.NET) 418
 - Capture 437
 - creating 429
 - Index 430
 - Length 430
 - Success 430
 - ToString 430
 - using 430
 - Value 430
- GroupCollection 429, 438
- groupCount method 377
- grouping and capturing 20-22
- grouping-only parentheses (see non-capturing parentheses)
- GroupNameFromNumber 427-428
- GroupNumberFromName 427-428
- Groups Match object method 429
- \p{Gujarati} 122
- Gutierrez, David xxiv
- \p{Han} 122
- hand tweaking
 - alternation 261
 - caveats 253
- \p{Hangul_Jamo} 124
- hasAnchoringBounds method 388
- HASH(0x80f60ac) 257
- hasTransparentBounds method 387
- Hazel, Philip xxiv, 91, 440
- \p{Hebrew} 122, 124
- height attribute Java example 397
- Hz 109
- hex escape 117-118
 - Perl 286
- highlighting with ANSI escape sequences 79
- \p{Hiragana} 122
- history
 - '\' 87
 - AT&T Bell Labs 86
 - awk* 87
 - Berkeley 86
 - ed* trivia 86
 - egrep* 86-87
 - grep* 86
 - lex* 87
 - Perl 88-90, 308
 - PHP 440
 - of regexes 85-91

- history (cont'd)
 - sed* 87
 - underscore in `\w` 89
 - `/x` 90
- hitEnd method 389-392
- hostname example 22, 73, 76, 98-99, 137-138, 203, 260, 267-268, 304, 306, 450-451
 - egrep* 25
 - Java 209
 - plucking from text 71-73, 206-208
 - in URL 74-77
 - validating 203-205
 - VB.NET 204
- `$HostnameRegex` 76, 137, 303, 351
- hot VM 236
- HREF example 452
- HTML
 - cooking 68, 414
 - matching tag 200-201
- HTML example 443-444, 459, 461, 464, 481, 484
 - conversion from text 67-77
 - cooking 68, 414
 - encoding 414
 - `<HR>` 194
 - link 201-203
 - optional 140
 - paired tags 165
 - parsing 132, 315, 321, 399
 - tag 9, 18-19, 26, 200-201, 326, 357
 - URL 74-77, 203, 206-208, 303, 450-451
 - URL-encoding 320
- `htmlspecialchars` 461
- HTTP newlines 115
- HTTP response example 467
- HTTP URL example 25, 74-77, 201-204, 206-209, 260, 303-304, 306, 320, 450-451
- `http://regex.info/` xxiii, 7, 345, 471
- `$HttpUrl` 303, 305, 345, 351
- hybrid regex engine 182, 239, 243
- hyphen in character class 9
- Hz 109

- (?i) (see: case-insensitive mode; mode modifier)
- /i 135
 - (see also: case-insensitive mode; mode modifier)
 - introduced 47
 - with `study` 359
- i as -y 86
- identifier matching 24
- if (see conditional)
- if vs. while vs. foreach 320
- (*? if then | else*) (see conditional)
- IgnoreCase (.NET) 96, 99, 408, 419, 427
- IgnorePatternWhitespace (.NET) 99, 408, 419, 427
- IllegalArgumentException 373, 380
- IllegalStateException 376-377
- image tags Java example 397
- image tags example 397
- implementation of engine 183
- implicit 362
- implicit anchor optimization 246
- Imports 413, 415, 434
 - `\p{InArrows}` 124
 - `\p{InBasic_Latin}` 124
 - `\p{InBox_Drawing}` 124
 - `\p{InCurrency}` 124
 - `\p{InCyrillic}` 124
- Index
 - Group object method 430
 - Match object method 429
- IndexOutOfBoundsException 375-376, 380
 - `\p{InDingbats}` 124
- indispensable TiVo 3
 - `\p{InHangul_Jamo}` 124
 - `\p{InHebrew}` 124
 - `\p{Inherited}` 122
- initial class discrimination 245-248, 252, 257-259, 332, 361
 - `\p{Initial_Punctuation}` 123
 - `\p{InKatakana}` 124
- inline modes (see modifiers)
 - `\p{InTamil}` 124
- integrated handling 94
 - compile caching 243
- interpolation 288-289
 - caching 351
 - introduced 77
 - mimicking 321
 - PHP 103
- INTERSECTION class set operations 126
- interval 141
 - introduced 20
 - `[X{0,0}]` 141
 - `\p{InTibetan}` 124
- introduced encoding 29
- introduction Perl 37-38
- IOException 81

IP example 5, 187-189, 267-268, 311, 314, 348-349

Iraq 11

Is vs. In 121, 124-125

- Java 369
- .NET 407
- Perl 288

`\p{IsCherokee}` 122

`\p{IsCommon}` 122

`\p{IsCyrillic}` 122

`\p{IsGujarati}` 122

`\p{IsHan}` 122

`\p{IsHebrew}` 122

`\p{IsHiragana}` 122

`isJavaIdentifierStart` 369

`\p{IsKatakana}` 122

`\p{IsLatin}` 122

IsMatch (Regex object method) 421

ISO-8859-1 encoding 29, 87, 106, 108, 123

issues overview encoding 105

`\p{IsThai}` 122

`\p{IsTibetan}` 124

ĵ 111

Japanese

- 正規表現は簡単だよ! 5
- text processing 29

“japhy” 246

Java 95-96, 365-403

- (see also `java.util.regex`)
- after-match data 138
- anchoring bounds 388
- benchmarking 235-236
- BLTN 236
- bugs 365, 368-369, 387, 392, 399, 403
- code example 81, 209, 217, 235, 371, 375, 378-379, 381-384, 389
- CSV parsing example 401
- description 365
- dot modes 111, 370
- doubled-word example 81
- JIT 236
- line anchors 130, 370, 388
- line terminators 370
- match modes 368
- match pointer 374, 383, 398, 400
- matching comments 272-276
- method chaining 389
- method index 366
- Mustang 401
- object model 371-372
- `\p{...}` 125

Java (cont'd)

- regex flavor 366-370
- region 384-389
- search and replace 378-383
- Σ 110
- split 395-396
- strings 102
- transparent bounds 387
- Unicode 369
- URL example 209
- version covered 365
- version history 365, 368-369, 392, 401
- VM 236
- word boundaries 134

java properties 369

`\p{javaJavaIdentifierStart}` 369

`java.lang.Character` 369

`java.util.regex` (see Java)

`java.util.Scanner` 390

Jeffs example 61-64

JfrieDlsRegexLibrary 434-435

JIT

- Java 236
- .NET 410

JRE 236

`\p{Katakana}` 122, 124

keeping in sync 210-211

Keisler, H. J. 85

Kleene, Stephen 85

The Kleene Symposium 85

`\kname` (see named capture)

Korean text processing 29

Kunen, K. 85

£ 124

`\l` 290

`\p{L&}` 122-123, 125, 442

- Java 369
- Perl 288

`\p{L}` 121-122, 133, 368, 395

language (see also: .NET; C#; Java; MySQL; Perl; procmail; Python; Ruby; Tcl; VB.NET)

- character class 10, 13
- identifiers 24

`\p{Latin}` 122

Latin-1 encoding 29, 87, 106, 108, 123

lazy 166-167

- (see also greedy)
- essence 159, 168-169

- lazy** (cont'd)
 - favors match 167-168
 - vs. greedy 169, 256-257
 - optimization 248, 257
 - quantifier 141
- lazy evaluation** 181, 355
- `\L` `\E` 290
 - inhibiting 292
- `lc` 290
- `lcfirst` 290
- leftmost match** 177-179
- `Length`
 - Group object method 430
 - Match object method 429
- length-cognizance optimization** 245, 247
 - `\p{Letter}` 122, 288
 - `\p{Letter_Number}` 123
 - `$LevelN` 330, 343
- lex** 86
 - `$` 112
 - `dot` 111
 - history 87
 - and trailing context 182
- lexer** 132, 389, 399
 - building 315
- lexical scope** 299
- `LF` 109, 370
- LIFO backtracking** 159
- limit**
 - backtracking 239
 - `preg_split` 466-467
 - recursion 249-250
- line** (see also string)
 - anchor optimization 246
 - vs. string 55
- line anchor** 112-113
 - mechanics of matching 150
 - variety of implementations 87
- line anchors**
 - Java 130, 370, 388
 - .NET 130
 - Perl 130
 - PHP 130
- line feed** 109, 370
- `LINE SEPARATOR` 109, 123, 370
- line terminators** 109-111, 129-130, 370
 - with `$` and `^` 112
 - Java 370
- `\p{Line_Separator}` 123
- link**
 - matching 201
 - (see also URL examples)
 - Java 209
- link, matching** (cont'd)
 - VB.NET 204
- list context** 294, 310-311
 - forcing 310
- literal string** initial string discrimination 245-248, 252, 257-259, 332, 361
- literal text**
 - exposing 255
 - introduced 5
 - mechanics of matching 149
 - pre-check optimization 245-248, 252, 257-259, 332, 361
- literal-text mode** 113, 136, 290
 - inhibiting 292
 - .NET 136
- `\p{Ll}` 123, 406
- `\p{Lm}` 123, 406
- `\p{Lo}` 123, 406
- `local` 296, 341
 - in embedded code 336
 - vs. `my` 297
- locale** 127-128
 - overview 87
 - `\w` 120-121
- localizing** 296-297
- `localtime` 294, 319, 351
- lock up** (see neverending match)
- locking in regex literal** 352
- "A logical calculus of the ideas imminent in nervous activity" 85
- longest match** finding 334-335
- longest-leftmost match** 148, 177-179
- lookahead** 133
 - (see also lookaround)
 - auto 410
 - introduced 60
 - mimic atomic grouping 174
 - mimic optimizations 258-259
 - negated
 - ` ..` 167
 - positive vs. negative 66
- lookahead example** 61-64
- lookaround**
 - backtracking 173-174
 - conditional 140-141
 - and DFAs 182
 - doesn't consume text 60
 - introduced 59
 - mimicking class set operations 126
 - mimicking word boundaries 134
 - Perl 288

- lookbehind** 133
 - (see also lookaround)
 - Java 368
 - .NET 408
 - Perl 288
 - PHP 134, 443
 - positive vs. negative 66
 - unlimited 408
- lookingAt method** 376
- loose matching** (see case-insensitive mode)
- Lord, Tom** 183
- \p{Lowercase_Letter}** 123
- LS** 109, 123, 370
- \p{Lt}** 123, 406
- \p{Lu}** 123, 406
- Lunde, Ken** xxiv, 29

- \p{M}** 120, 122
- m/ /** introduced 38
- (?m)** (see: enhanced line-anchor mode; mode modifier)
- /m** 135
 - (see also: enhanced line-anchor mode; mode modifier)
- machine-dependent character codes** 115
- MacOS** 115
- mail processing example** 53-59
- makudonarudo example** 165, 169, 228-232, 264
- \p{Mark}** 122
- match** 306-318
 - (see also: DFA; NFA)
 - actions 95
 - context 294-295, 309
 - list 294, 310-311
 - scalar 294, 310, 312-316
 - DFA vs. NFA 224
 - efficiency 179
 - example with backtracking 160
 - example without backtracking 160
 - lazy example 161
 - leftmost-longest 335
 - longest 334-335
 - m/ /**
 - introduced 38
 - mechanics (see also: greedy; lazy)
 - . * 152
 - anchors 150
 - capturing parentheses 149
 - character classes and dot 149
 - consequences 156
- match, mechanics** (cont'd)
 - greedy introduced 151
 - literal text 149
 - modes 110-113
 - Java 368
 - negating 309
 - neverending 222-228, 330, 340
 - avoiding 264-266
 - discovery 226-228
 - explanation 226-228
 - non-determinism 264
 - short-circuiting 250
 - solving with atomic grouping 268
 - solving with possessive quantifiers 268
 - NFA vs. DFA 156-157, 180-183
 - of nothing 454
 - position (see pos)
 - POSIX
 - Perl 335
 - shortest-leftmost 182
 - side effects 317
 - intertwined 43
 - Perl 40
 - speed 181
 - in a string 27
 - tag-team 132
 - viewing mechanics 331-332
- Match Empty** 433
- match modes** Java 368
- Match(.NET) Success** 96
- Match object (.NET)** 417
 - Capture** 437
 - creating** 421, 429
 - Groups** 429
 - Index** 429
 - Length** 429
 - NextMatch** 429
 - Result** 429
 - Success** 427
 - Synchronized** 430
 - ToString** 427
 - using** 427
 - Value** 427
- match pointer** Java 374, 383, 398, 400
- Match (Regex object method)** 421
- “match rejected by optimizer”** 363
- match results** Java 376
- MatchCollection** 422
- Matcher**
 - appendReplacement** 380
 - appendTail** 381
 - end** 377

- Matcher (cont'd)
 - find 375
 - group 377
 - groupCount 377
 - hasAnchoringBounds 388
 - hasTransparentBounds 387
 - hitEnd 389-392
 - lookingAt 376
 - matches 376
 - pattern 393
 - quoteReplacement 379
 - region 384-389
 - region 386
 - regionEnd 386
 - regionStart 386
 - replaceAll 378
 - replaceFirst 379
 - replacement argument 380
 - requireEnd 389-392
 - reset 392-393
 - start 377
 - text 394
 - toMatchResult 377
 - toString 393
 - useAnchoringBounds 388
 - usePattern 393, 399
 - useTransparentBounds 387
- Matcher object 373
 - reusing 392-393
- \$matches 450
 - vs. \$all_matches 454
- matches
 - unexpected 194-195
 - viewing all 332
- matches method 376, 395
- Matches (Regex object method) 422
- MatchEvaluator 423-424
- matching
 - delimited text 196-198
 - HTML tag 200
 - longest-leftmost 177-179
- matching comments Java 272-276
- MatchObject object (.NET) creating 422
- \p{Math_Symbol} 123
- Maton, William xxiv, 36
- mb_ereg suite 439
- MBOL 362
- \p{Mc} 123
- McCloskey, Mike xxiv
- McCulloch, Warren 85
- \p{Me} 123
- mechanics viewing 331-332
- metacharacter
 - conflicting 44-46
 - differing contexts 10
 - first-class 87, 92
 - introduced 5
 - vs. metasequence 27
- metasequence defined 27
- method chaining 389
 - Java 389
- method index Java 366
- mimic
 - \$' 357
 - \$` 357
 - && 302, 357
 - atomic grouping 174
 - class set operations 126
 - conditional with lookahead 140
 - initial-character discrimination optimization 258-259
 - named capture 344-345
 - POSIX matching 335
 - possessive quantifiers 343-344
 - variable interpolation 321
 - word boundaries 66, 134, 341-342
- minlen *length* 362
- minus in character class 9
- MISL .NET 410
- "missing" functions PHP 471
- \p{Mn} 123
- mode modifier 110, 135-136
- mode-modified span 110, 135-136, 367, 392, 407, 446
- modes introduced with *egrep* 14-15
- \p{Modifier_Letter} 123
- modifiers 372
 - (see also match, modes)
 - combining 69
 - example with five 316
 - /g 51
 - /i 47
 - "locking in" 304-305
 - notation 99
 - /osmosis 293
 - Perl 292-293
 - Perl core 292-293
 - with regex object 304-305
 - unknown 448
- \p{Modifier_Symbol} 123
- Morse, Ian xxiv
- motto Perl 348
- Mre=debug (see use re 'debug')
- multi-character quotes 165-166
- Multiline (.NET) 408, 419-420, 427

- multiple-byte character encoding** 29
- `MungeRegexLiteral` 342-344, 346
- Mustang** Java 401
- my**
 - binding 339
 - in embedded code 338-339
 - vs. `local` 297
- MySQL**
 - after-match data 138
 - `DBIx::DWIW` 258
 - version covered 91
 - word boundaries 134
-
- `\p{N}` 122, 395
- `\n` 49, 115-116
 - introduced 44
 - machine-dependency 115
- `$^N` 300-301, 344-346
- `(?n)` 408
- named capture** 138
 - mimicking 344-345
 - `.NET` 408-409
 - numeric names 451
 - PHP 450-452, 457, 476-477
 - with unnamed capture 409
- naughty variables** 356
 - OK for debugging 331
- `\p{Nd}` 123, 368, 406
- negated class**
 - introduced 10-11
 - and lazy quantifiers 167
 - Tcl 112
- negative lookahead** (see `lookahead`, `negative`)
- negative lookbehind** (see `lookbehind`, `negative`)
- `NEL` 109, 370, 407
- nervous system** 85
- nested constructs**
 - `.NET` 436
 - Perl 328-331, 340-341
 - PHP 475-478, 481
- `$NestedStuffRegex` 339, 346
- `.NET` xvii, 405-438
 - `+` 202
 - after-match data 138
 - benchmarking 237
 - character-class subtraction 406
 - code example 219
 - flavor overview 92
 - JIT 410
 - line anchors 130
- `.NET` (cont'd)
 - literal-text mode 136
 - MISL 410
 - object model 417
 - `\p{...}` 125
 - regex approach 96-97
 - regex flavor 407
 - search and replace 414, 423-424
 - URL example 204
 - version covered 405
 - word boundaries 134
 - (see also `VB.NET`)
- neurophysiologists** early regex study 85
- neverending match** 222-228, 330, 340
 - avoiding 264-266
 - discovery 226-228
 - explanation 226-228
 - non-determinism 264
 - short-circuiting 250
 - solving with atomic grouping 268
 - solving with possessive quantifiers 268
- `New Regex` 96, 99, 416, 421
- newline** and HTTP 115
- `NEXT LINE` 109, 370, 407
- `NextMatch` (`Match` object method) 429
- NFA**
 - acronym spelled out 156
 - and alternation 174-175
 - compared with DFA 156-157, 180-183
 - control benefits 155
 - efficiency 179
 - essence (see `backtracking`)
 - first introduced 145
 - freeflowing regex 277-281
 - and greediness 162
 - implementation ease 183
 - introduction 153
 - nondeterminism 265
 - checkpoint 264-265
 - POSIX efficiency 179
 - testing for 146-147
 - theory 180
- `\p{NL}` 123
- `\N{LATIN SMALL LETTER SHARP S}` 290
- `\N{name}` 290
 - (see also `pragma`)
 - inhibiting 292
- `\p{No}` 123
- No Dashes Hall Of Shame** 458
- `no re 'debug'` 361
- `no_match_vars` 357
- nomenclature** 27

- non-capturing parentheses 45, 137-138
 - (see also parentheses)
- Nondeterministic Finite Automaton (see NFA)
- None (.NET) 421, 427
- non-greedy (see lazy)
- nonillion 226
- nonparticipation parentheses 450, 453-454, 469
- nonregular sets 180
- \p{Non_Spacing_Mark} 123
- non-word boundaries (see word boundaries)
- “normal” 263-266
- NUL 117
 - with dot 119
- NULL 454
- \p{Number} 122

- /o 352-353
 - with regex object 354
- Obfuscated Perl Contest 320
- object model
 - Java 371-372
 - .NET 416-417
- Object Oriented Perl* 339
- object-oriented handling 95-97
 - compile caching 244
- octal escape 116, 118
 - vs. backreference 412-413
 - Perl 286
- offset preg_match 453
- on-demand recompilation 351
- onemself example 332, 334
- \p{Open_Punctuation} 123
- operators Perl list 285
- optimization 240-252
 - (see also: atomic grouping; possessive quantifiers; efficiency)
 - automatic possessification 251
 - BLTN 236
 - with bump-along 255
 - end-of-string anchor 246
 - excessive backtrack 249-250
 - hand tweaking 252-261
 - implicit line anchor 191
 - initial character discrimination 245-248, 252, 257-259, 332, 361
 - JIT 236, 410
 - lazy evaluation 181
 - lazy quantifier 248, 257
 - leading |.*| 246
- optimization (cont'd)
 - literal-string concatenation 247
 - need cognizance 252
 - needless class elimination 248
 - needless parentheses 248
 - pre-check of required character 245-248, 252, 257-259, 332, 361
 - simple repetition
 - discussed 247-248
 - small quantifier equivalence 251-252
 - state suppression 250-251
 - string/line anchors 149, 181
 - super-linear short-circuiting 250
- option
 - o 36
 - c 361
 - Dr 363
 - e 36, 53, 361
 - i 53
 - M 361
 - Mre=debug 363
 - n 36
 - p 53
 - w 38, 296, 326, 361
- Option (.NET) 415
- optional (see also quantifier)
 - whitespace 18
- Options (Regex object method) 427
- OR class set operations 125-126
- Oram, Andy 5
- ordered alternation 175-177
 - (see also alternation, ordered)
 - pitfalls 176
- osmosis 293
- /osmosis 293
- \p{Other} 122
- \p{Other_Letter} 123
- \p{Other_Number} 123
- \p{Other_Punctuation} 123
- \p{Other_Symbol} 123
- our 295, 336
- overload pragma 342

- \p{...}
 - Java 125
 - .NET 125
 - PHP 125
- \p{P} 122
- \p{^...} 288
- \p{A11} 125
 - Perl 288

- `\p{all}` 369
- `panic: top_env` 332
- `\p{Any}` 125, 442
 - Perl 288
- Papen, Jeffrey** xxiv
- PARAGRAPH SEPARATOR** 109, 123, 370
- `\p{Paragraph_Separator}` 123
- parentheses**
 - as `\(·\)` 86
 - and alternation 13
 - balanced 328-331, 340-341, 436, 475-478, 481
 - difficulty 193-194
 - capturing 137, 300
 - and DFAs 150, 182
 - introduced with *egrep* 20-22
 - mechanics 149
 - Perl 41
 - capturing only 152
 - counting 21
 - elimination optimization 248
 - grouping-only (see non-capturing parentheses)
 - limiting scope 18
 - named capture 138, 344-345, 408-409, 450-452, 457, 476-477
 - nested 328-331, 340-341, 436, 475-477, 481
 - non-capturing 45, 137-138
 - non-participating 300
 - nonparticipation 450, 453-454, 469
 - with split
 - .NET 409, 426
 - Perl 326
- `\p{Arrows}` 124
- parser** 132, 389, 399
- parsing** regex 410
- participate in match** 140
- Pascal** 36, 59, 183
 - matching comments of 265
- `\p{Assigned}` 125-126
 - Perl 288
- patch** 88
- path** (see backtracking)
- pathname example** 190-192
- Pattern**
 - CANON_EQ 108, 368
 - CASE_INSENSITIVE 95, 110, 368, 372
 - CASE_INSENSITIVE bug 392
 - COMMENTS 99, 219, 368, 401
 - compile 372
 - DOTALL 368, 370
 - flags 394
- Pattern (cont'd)**
 - matcher 373
 - matches 395
 - MULTILINE 81, 368, 370
 - MULTILINE bug 387
 - pattern 394
 - quote 395
 - split 395-396
 - toString 394
 - UNICODE_CASE 368, 372
 - UNIX_LINES 368, 370
- pattern argument** 472
 - array order 462, 464
- pattern arguments** PHP 444, 448
- pattern method** 393-394
- pattern modifier**
 - A 447
 - D 442, 447
 - e 459, 465, 478
 - m 442
 - S 259, 447, 460, 467, 478-480
 - u 442, 447-448, 452-453
 - U 447
 - unknown errors 448
 - x 443, 471
 - X 447
- pattern modifiers** PHP 446-448
- PatternSyntaxException** 371, 373
- `\p{Basic_Latin}` 124
- `\p{Box_Drawing}` 124
- `\p{C}` 122
 - Java 369
- `\p{Pc}` 123, 406
- `\p{Cc}` 123
- `\p{Cf}` 123
- `\p{Cherokee}` 122
- `\p{Close_Punctuation}` 123
- `\p{Cn}` 123, 125-126, 369, 408
 - Java 369
- `\p{Co}` 123
- `\p{Connector_Punctuation}` 123
- `\p{Control}` 123
- PCRE** 91, 440
 - (see also PHP)
 - “extra stuff” 447
 - flavor overview 441
 - lookbehind 134
 - recursive matching 475-478
 - study 447
 - version covered 440
 - `\w` 120
 - web site 91
 - X pattern modifier 447

pcre_study 259
 \p{Currency} 124
 \p{Currency_Symbol} 123
 \p{Cyrillic} 122, 124
 \p{Pd} 123
 \p{Dash_Punctuation} 123
 \p{Decimal_Digit_Number} 123
 \p{Dingbats} 124
 \p{Pe} 123
PeakWebbosting.com xxiv
 \p{Enclosing_Mark} 123
people
 Aho, Alfred 86, 180
 Barwise, J. 85
 Byington, Ryan xxiv
 Click, Cliff xxiv
 Constable, Robert 85
 Conway, Damian 339
 Cruise, Tom 51
 Filo, David 397
 Fite, Liz 33
 Friedl, Alfred 176
 Friedl, brothers 33
 Friedl, Fumie v, xxiv
 birthday 11-12
 Friedl, Jeffrey xxiii
 Friedl, Stephen xxiv, 458
 George, Kit xxiv
 Gill, Stuart xxiv
 Gosling, James 89
 Greant, Zak xxiv
 Gutierrez, David xxiv
 Hazel, Philip xxiv, 91, 440
 Keisler, H. J. 85
 Kleene, Stephen 85
 Kunen, K. 85
 Lord, Tom 183
 Lunde, Ken xxiv, 29
 Maton, William xxiv, 36
 McCloskey, Mike xxiv
 McCulloch, Warren 85
 Morse, Ian xxiv
 Oram, Andy 5
 Papen, Jeffrey xxiv
 Perl Porters 90
 Pinyan, Jeff 246
 Pitts, Walter 85
 Reinhold, Mark xxiv
 Sethi, Ravi 180
 Spencer, Henry 88, 182-183, 243
 Thompson, Ken 85-86, 111
 Tubby 265
 Ullman, Jeffrey 180

people (cont'd)
 Wall, Larry 88-90, 140, 363
 Zawodny, Jeremy 258
 Zmievski, Andrei xxiv, 440
Perl
 \p{...} 125
 \$/ 35
 context (see also match, context)
 contorting 294
 efficiency 347-363
 flavor overview 92, 287
 greatest weakness 286
 history 88-90, 308
 introduction 37-38
 line anchors 130
 modifiers 292-293
 motto 348
 option
 -o 36
 -c 361
 -Dr 363
 -e 36, 53, 361
 -i 53
 -M 361
 -Mre=debug 363
 -n 36
 -p 53
 -w 38, 296, 326, 361
 regex operators 285
 search and replace 318-321
 Σ 110
 Unicode 288
 version covered 283
 warnings 38
 (\$^w variable) 297
 use warnings 326, 363
Perl Porters 90
perladmin 299
 \p{Pf} 123
 Java 369
 \p{Final_Punctuation} 123
 \p{Format} 123
 \p{Gujarati} 122
 \p{Han} 122
 \p{Hangul_Jamo} 124
 \p{Hebrew} 122, 124
 \p{Hiragana} 122
PHP 439-484
 after-match data 138
 benchmarking 234-235
 callback 463, 465
 CSV parsing example 480
 efficiency 478-480

- PHP (cont'd)
- flavor overview 441
 - history 440
 - line anchors 130
 - lookbehind 134, 443
 - “missing” functions 471
 - `\p{...}` 125
 - pattern arguments 444, 448
 - recursive matching 475-478
 - regex delimiters 445, 448
 - search and replace 458-465
 - single-quoted string 444
 - strings 103-104
 - `str_replace` 458
 - study 447
 - Unicode 442, 447
 - version covered 440
 - `\w` 120
 - word boundaries 134
 - `\p{Pi}` 123
 - Java 369
 - `\p{InArrows}` 124
 - `\p{InBasic_Latin}` 124
 - `\p{InBox_Drawing}` 124
 - `\p{InCurrency}` 124
 - `\p{InCyrillic}` 124
 - `\p{InDingbats}` 124
 - `\p{InHangul_Jamo}` 124
 - `\p{InHebrew}` 124
 - `\p{Inherited}` 122
 - `\p{Initial_Punctuation}` 123
 - `\p{InKatakana}` 124
 - `\p{InTamil}` 124
 - `\p{InTibetan}` 124
 - Pinyan, Jeff 246
 - `\p{IsCherokee}` 122
 - `\p{IsCommon}` 122
 - `\p{IsCyrillic}` 122
 - `\p{IsGujarati}` 122
 - `\p{IsHan}` 122
 - `\p{IsHebrew}` 122
 - `\p{IsHiragana}` 122
 - `\p{IsKatakana}` 122
 - `\p{IsLatin}` 122
 - `\p{IsThai}` 122
 - `\p{IsTibetan}` 124
 - Pitts, Walter 85
 - `\p{javaJavaIdentifierStart}` 369
 - `\p{Katakana}` 122, 124
 - `\p{L}` 121-122, 133, 368, 395
 - `\p{L&}` 122-123, 125, 442
 - Java 369
 - Perl 288
 - `\p{L PHP}` 442
 - `\p{Latin}` 122
 - `(?P<->)` 451-452, 457
 - `(?P<name>...)` (see named capture)
 - `\p{Letter}` 122, 288
 - `\p{Letter_Number}` 123
 - `\p{Line_Separator}` 123
 - `\p{Ll}` 123, 406
 - `\p{Lm}` 123, 406
 - `\p{Lo}` 123, 406
 - `\p{Lowercase_Letter}` 123
 - `\p{Lt}` 123, 406
 - `\p{Lu}` 123, 406
 - plus
 - as `\+` 141
 - backtracking 162
 - greedy 141, 447
 - introduced 18-20
 - lazy 141
 - possessive 142
 - `\p{M}` 120, 122
 - `\p{Mark}` 122
 - `\p{Math_Symbol}` 123
 - `\p{Mc}` 123
 - `\p{Me}` 123
 - `\p{Mn}` 123
 - `\p{Modifier_Letter}` 123
 - `\p{Modifier_Symbol}` 123
 - `\p{N}` PHP 442
 - `\p{N}` 122, 395
 - `(?P=name...)` (see named capture)
 - `\p{Nl}` 123, 368, 406
 - `\p{Nl}` 123
 - `\p{No}` 123
 - `\p{Non_Spacing_Mark}` 123
 - `\p{Number}` 122
 - `\p{Po}` 123
 - `\p{Open_Punctuation}` 123
 - population example 59
 - `pos` 130-133, 313-314, 316
 - (see also `\G`)
 - positive lookahead (see lookahead, positive)
 - positive lookbehind (see lookbehind, positive)
 - POSIX
 - `[...]` 128
 - `[:::]` 127
 - Basic Regular Expressions 87-88
 - bracket expressions 127
 - character class 127
 - character class and locale 127
 - character equivalent 128

- POSIX (cont'd)
 - collating sequences 128
 - dot 119
 - empty alternatives 140
 - Extended Regular Expressions 87-88
 - superficial flavor chart 88
 - locale 127
 - overview 87
 - longest-leftmost rule 177-179, 335
- POSIX NFA
 - backtracking example 229
 - testing for 146-147
- possessive quantifier 477, 483
- possessive quantifiers 142, 172-173, 477, 483
 - (see also atomic grouping)
 - automatic 251
 - for efficiency 259-260, 268-270, 482
 - mimicking 343-344
 - optimization 250-251
- possessive quantifiers example 198, 201
- postal code example 209-212
 - \p{Other} 122
 - \p{Other_Letter} 123
 - \p{Other_Number} 123
 - \p{Other_Punctuation} 123
 - \p{Other_Symbol} 123
 - £ 124
 - \p{P} 122
 - \p{Paragraph_Separator} 123
 - \p{Pc} 123, 406
 - \p{Pd} 123
 - \p{Pe} 123
 - \p{Pf} 123
 - Java 369
 - \p{Pi} 123
 - Java 369
 - \p{Po} 123
 - \p{Private_Use} 123
 - \p{Ps} 123
 - \p{Punctuation} 122
- pragma
 - charnames 290
 - (see also \N{name})
 - overload 342
 - re 361, 363
 - strict 295, 336, 345
 - warnings 326, 363
- pre-check of required character 245-248, 252, 257-259, 361
 - mimic 258-259
 - viewing 332
- preg function interface 443-448
- preg suite 439
 - “missing” functions 471
- preg_grep 469-470
- PREG_GREP_INVERT 470
- preg_match 449-453
 - offset 453
- preg_match_all 453-457
- PREG_OFFSET_CAPTURE 452, 454, 456
- preg_pattern_error 474
- PREG_PATTERN_ORDER 455
- preg_quote 136, 470-471
- preg_regex_error 475
- preg_regex_to_pattern 472-474
- preg_replace 458-464
- preg_replace_callback 463-465
- PREG_SET_ORDER 456
- preg_split 465-469
 - PREG_SPLIT_DELIM_CAPTURE 468-469
 - split limit 469
- PREG_SPLIT_NO_EMPTY 468
- PREG_SPLIT_OFFSET_CAPTURE 468
- pre-match copy 355
- prepending filename to line 79
- price rounding example 51-52, 167-168
 - with alternation 175
 - with atomic grouping 170
 - with possessive quantifier 169
- Principles of Compiler Design* 180
- printf 40
- private vs. global Perl variables 295
- \p{Private_Use} 123
- procedural handling 95-97
 - compile caching 244
- processing instructions 483
- procmail 94
 - version covered 91
- Programming Perl* 283, 286, 339
- promote 294-295
- properties 121-123, 125-126, 288, 368-369, 442
- PS 109, 123, 370
- \p{S} 122
- \p{Ps} 123
- \p{Sc} 123-124
- \p{Separator} 122
- \p{Sk} 123
- \p{Sm} 123
- \p{So} 123
- \p{Space_Separator} 123
- \p{Spacing_Combining_Mark} 123
- \p{Symbol} 122
- \p{Tamil} 124
- \p{Thai} 122

- `\p{Tibetan}` 124
- `\p{Titlecase_Letter}` 123
- publication**
 - Bulletin of Math. Biophysics* 85
 - CJKV Information Processing* 29
 - Communications of the ACM* 85
 - Compilers—Principles, Techniques, and Tools* 180
 - Embodiments of Mind* 85
 - The Kleene Symposium* 85
 - “A logical calculus of the ideas imminent in nervous activity” 85
 - Object Oriented Perl* 339
 - Principles of Compiler Design* 180
 - Programming Perl* 283, 286, 339
 - Regular Expression Search Algorithm* 85
 - “The Role of Finite Automata in the Development of Modern Computing Theory” 85
- `\p{Unassigned}` 123, 125
 - Perl 288
- `\p{Punctuation}` 122
- `\p{Uppercase_Letter}` 123
- Python**
 - after-match data 138
 - benchmarking 238-239
 - line anchors 130
 - mode modifiers 135
 - regex approach 97
 - strings 104
 - version covered 91
 - word boundaries 134
 - `\z` 112
- `\p{Z}` 121-122, 368, 407
- `\pZ` PHP 442
- `\p{Zl}` 123
- `\p{Zp}` 123
- `\p{Zs}` 123

- `\Q` Java 368, 395, 403
- Qantas** 11
- `\Q·\E` 290
 - inhibiting 292
- qed** 85
- `qr/.../` (see also regex objects)
 - introduced 76
- quantifier** (see also: plus; star; question mark; interval; lazy; greedy; possessive quantifiers)
 - and backtracking 162
 - factor out 255
 - grouping for 18
- quantifier** (cont'd)
 - multiple levels 266
 - optimization 247-248
 - and parentheses 18
 - possessive 477, 483
 - possessive quantifiers 142, 172-173, 477, 483
 - for efficiency 259-260, 268-270, 482
 - automatic optimization mimicking
 - question mark
 - as `\?` 141
 - backtracking 160
 - greedy 141, 447
 - introduced 17-18
 - lazy 141
 - possessive 142
 - smallest preceding subexpression 29
- question mark**
 - as `\?` 141
 - backtracking 160
 - greedy 141, 447
 - introduced 17-18
 - lazy 141
 - possessive 142
- quote method** 136, 395
- quoted string** (see double-quoted string example)
- quoteReplacement method** 379
- quotes** multi-character 165-166

- `r"..."` 104
- `\r` 49, 115-116
 - machine-dependency 115
- `(?R)` 475
 - PCRE 475
 - PHP 475
- `$$R` 302, 327
- `re` 361, 363
- `re pragma` 361, 363
- reality check** 226-228
- recursive matching** (see also dynamic regex)
 - Java 402
 - .NET 436
 - PCRE 475-478
 - PHP 475-478, 481-484
- red dragon** 180
- Reflection** 435

regex

- balancing needs 186
- cache 242-245, 350-352, 432, 478
- compile 179-180, 350
- default 308
- delimiters 291-292
- DFA (see DFA)
- encapsulation (see regex objects)
- engine analogy 143-147
- vs. English 275
- error checking 474
- frame of mind 6
- freeflowing design 277-281
- history 85-91
- library 76, 208
- longest-leftmost match 177-179
 - shortest-leftmost 182
- mechanics 241-242
- NFA (see NFA)
- nomenclature 27
- operands 288-292
- overloading 291, 328
 - inhibiting 292
 - problems 344
- subexpression
 - defined 29
- subroutines 476

regex approach .NET 96-97

regex delimiters PHP 445, 448

regex flavor

- Java 366-370
- .NET 407

regex literal 288-292, 307

- inhibiting processing 292
- locking in 352
- parsing of 292
- processing 350
- regex objects 354

Regex (.NET)

- CompileToAssembly 433, 435
- creating
 - options 419-421
- Escape 432
- GetGroupNames 427-428
- GetGroupNumbers 427-428
- GroupNameFromNumber 427-428
- GroupNumberFromName 427-428
- IsMatch 413, 421, 431
- Match 96, 414, 416, 421, 431
- Matches 422, 431
- object
 - creating 96, 416, 419-421
 - exceptions 419

Regex (.NET), object (cont'd)

- using 96, 421
- Options 427
- Replace 414-415, 423-424, 431
- RightToLeft 427
- Split 425-426, 431
- ToString 427
- Unescape 433

regex objects 303-306

- (see also `qr/.../`)
- efficiency 353-354
- /g 354
- match modes 304-305
- /o 354
- in regex literal 354
- viewing 305-306

regex operators Perl 285

regex overloading 292

- (see also `use overload`)

regex overloading example 341-345

<http://regex.info/> xxiv, 7, 345, 358, 451

RegexCompilationInfo 435

regex-directed matching 153

- (see also NFA)
- and backreferences 303
- and greediness 162

Regex.Escape 136

RegexOptions

- Compiled 237, 408, 410, 420, 427-428, 435
- ECMAScript 406, 408, 412-413, 421, 427
- ExplicitCapture 408, 420, 427
- IgnoreCase 96, 99, 408, 419, 427
- IgnorePatternWhitespace 99, 408, 419, 427
- Multiline 408, 419-420, 427
- None 421, 427
- RightToLeft 408, 411-412, 420, 426-427, 429-430
- Singleline 408, 420, 427

region

- additional example 398
- anchoring bounds 388
- hitEnd 390
- Java 384-389
- methods that reset 385
- requireEnd 390
- resetting 392-393
- setting one edge 386
- transparent bounds 387

region method 386

regionEnd method 386

regionStart method 386

- reg_match 454
- regsub 100
- regular expression origin of term 85
- Regular Expression Search Algorithm* 85
- regular sets 85
- Reinhold, Mark xxiv
- removing whitespace 199-200
- Replace (Regex object method) 423-424
- replaceAll method 378
- replaceFirst method 379
- replacement argument 460
 - array order 462, 464
 - Java 380
 - PHP 459
- reproductive organs 5
- required character pre-check 245-248, 252, 257-259, 332, 361
- requireEnd method 389-392
- re-search-forward 100-101
- reset method 385, 392-393
- Result (Match object method) 429
- RightToLeft (Regex property) 427-428
- RightToLeft (.NET) 408, 411-412, 420, 426-427, 429-430
- “The Role of Finite Automata in the Development of Modern Computing Theory” 85
- Ruby
 - \$ and ^ 112
 - after-match data 138
 - benchmarking 238
 - line anchors 130
 - mode modifiers 135
 - version covered 91
 - word boundaries 134
- rule
 - earliest match wins 148-149
 - standard quantifiers are greedy 151-153
- rx 183

- \p{S} 122
- s/.../.../ 50, 318-321
- \s 49, 121
 - Emacs 128
 - introduction 47
 - Perl 288
 - PHP 442
- (?s) (see: dot-matches-all mode; mode modifier)
- \S 49, 56, 121
- /s 135

- /s (cont'd)
 - (see also: dot-matches-all mode; mode modifier)
- saved states (see backtracking, saved states)
- SawAmpersand 358
- say what you mean 195, 274
- SBOL 362
- \p{Sc} 123-124
- scalar context 294, 310, 312-316
 - forcing 310
- scanner 132, 389, 399
- schaffkopf 33
- scope lexical vs. dynamic 299
- scripts 122, 288, 442
- search and replace xvii
 - awk 100
 - Java 378-383
 - .NET 414, 423-424
 - Perl 318-321
 - PHP 458-465
 - Tcl 100
 - (see also substitution)
- sed
 - after-match data 138
 - dot 111
 - history 87
 - version covered 91
 - word boundaries 134
- 正規表現は簡単だよ! 5
- self-closing tag 481
- \p{Separator} 122
- server VM 236
- set operations (see class, set operations)
- Sethi, Ravi 180
- shell 7
- Σ 110
 - Java 110
 - Perl 110
- simple quantifier optimization 247-248
- single quotes delimiter 292, 319
- Singleline (.NET) 408, 420, 427
- single-quoted string PHP 444
- \p{Sk} 123
- \p{Sm} 123
- small quantifier equivalence 251-252
- \p{So} 123
- \p{Space_Separator} 123
- \p{Spacing_Combining_Mark} 123
- span (see: mode-modified span; literal-text mode)
- “special” 263-266
- Spencer, Henry 88, 182-183, 243

split

- with capturing parentheses
 - .NET 409, 426
 - Perl 326
 - PHP 468
- chunk limit
 - Java 396
 - Perl 323
 - PHP 466
- into characters 322
- Java 395-396
- limit 466-467
 - Java 396
 - Perl 323
 - PHP 466
- Perl 321-326
- PHP 465-469
- trailing empty items 324, 468
- whitespace 325

split method 395-396

Split (Regex object method) 425-426

ß 111, 128, 290

stacked data 456

standard formula for matching delimited text 196

star

- backtracking 162
- greedy 141, 447
- introduced 18-20
- lazy 141
- possessive 142

start method 377

start of match (see `\G`)

start of word (see word boundaries)

start-of-line/string (see anchor, caret)

start-of-string anchor optimization 246, 255-256, 315

states (see also backtracking, saved states)

- flushing (see: atomic grouping; look-around; possessive quantifiers)

stdClass *'list'* 362

stock pricing example 51-52, 167-168

- with alternation 175
- with atomic grouping 170
- with possessive quantifier 169

Strict (Option) 415

strict pragma 295, 336, 345

String

- matches 376
- replaceAll 378
- replaceFirst 379
- split 395

string (see also line)

- double-quoted (see double-quoted string example)
- initial string discrimination 245-248, 252, 257-259, 332, 361
- vs. line 55
- match position (see pos)
- pos (see pos)

StringBuffer 373, 380, 382, 397

StringBuilder 373, 382, 397

strings

- C# 103
- Emacs 101
- Java 102
- PHP 103-104
- Python 104
- as regex 101-105, 305
- Tcl 104
- VB.NET 103

stripping whitespace 199-200

str_replace 458

- PHP 458

study PHP 447

study 359-360

- when not to use 359

subexpression defined 29

subroutines regex 476

substitution xvii

- delimiter 319
- `s/.../.../` 50, 318-321
- (see also search and replace)

substring initial substring discrimination 245-248, 252, 257-259, 332, 361

subtraction

- character class 406
- class (set) 126
- class (simple) 125

Success

- Group object method 430
- Match object method 427

Sun's regex package (see `java.util.regex`)

super-linear (see neverending match)

super-linear short-circuiting 250

`\p{Symbol}` 122

Synchronized Match object method 430

syntax class Emacs 128

System.currentTimeMillis() 236

System.Reflection 435

System.Text.RegularExpressions 413, 415

- `\t` 49, 115-116
 - introduced 44
 - tag**
 - matching 200-201
 - XML 481
 - tag-team matching** 132, 315
 - `\p{Tamil}` 124
 - Tcl**
 - `[:<:]` 91
 - benchmarking 239
 - dot 111, 113
 - flavor overview 92
 - hand-tweaking 243, 259
 - line anchors 113, 130
 - mode modifiers 135
 - regex implementation 183
 - regsub 100
 - search and replace 100
 - strings 104
 - version covered 91
 - word boundaries 134
 - temperature conversion example**
 - Java 382
 - .NET 425
 - Perl 37, 283
 - PHP 444
 - terminators** (see line terminators)
 - testing engine type** 146-147
 - text method** 394
 - text-directed matching** 153
 - (see also DFA)
 - regex appearance 162
 - text-to-HTML example** 67-77
 - `\p{Thai}` 122
 - then** (see conditional)
 - theory of an NFA** 180
 - There's more than one way to do it* 349
 - this|that example** 133, 139, 243, 245-247, 252, 255, 260-261
 - Thompson, Ken** 85-86, 111
 - thread scheduling** Java benchmarking 236
 - `\p{Tibetan}` 124
 - tied variables** 299
 - `time()` 232
 - time of day** 26
 - `Time::HiRes` 232, 358, 360
 - `Time.new` 238
 - `Timer()` 237
 - timezone** PHP 235
 - title case** 110
 - `\p{Titlecase_Letter}` 123
 - TiVo** 3
 - tokenizer** 132, 389, 399
 - building 315
 - toMatchResult method** 377
 - toothpicks** scattered 101
 - tortilla** 128
 - ToString**
 - Group object method 430
 - Match object method 427
 - Regex object method 427
 - toString method** 393-394
 - Traditional NFA** testing for 146-147
 - trailing context** 182
 - transmission** (see also `\G`)
 - optimizations 246-247
 - transparent bounds** 387
 - Java 387
 - Tubby** 265
 - typographical conventions** xxi
-
- `\u` 117, 290, 406
 - `\U` 117
 - `\U...\E` 290
 - inhibiting 292
 - `uc` 290
 - `U+COB5` 107
 - `ucfirst` 290
 - UCS-2 encoding** 107
 - UCS-4 encoding** 107
 - Ullman, Jeffrey** 180
 - `\p{Unassigned}` 123, 125
 - Perl 288
 - unconditional caching** 350
 - underscore in `\w` history** 89
 - Unescape** 433
 - Unicode**
 - block 124
 - Java 369, 402
 - .NET 407
 - Perl 288
 - categories (see Unicode, properties)
 - character
 - combining 107, 120, 122
 - code point
 - beyond `U+FFFF` 109
 - introduced 107
 - multiple 108
 - unassigned in block 124
 - combining character 107, 120, 122
 - Java 368-369, 402-403
 - line terminators 109-111, 370
 - Java 370

- Unicode (cont'd)
 - loose matching (see case-insensitive mode)
 - .NET 407
 - official web site 127
 - overview 106-110
 - Perl 288
 - PHP 442, 447
 - properties 121, 369
 - (see also `\p{...}`)
 - Java 368
 - list 122-123
 - `\p{A11}` 125, 288
 - `\p{Any}` 125, 288, 442
 - `\p{Assigned}` 125-126, 288
 - Perl 288
 - PHP 442
 - `\p{Unassigned}` 123, 125, 288
 - script 122
 - Perl 288
 - PHP 442
 - Version 3.1 109
 - `\w` 120
 - whitespace and `/x` 288
- UnicodeData.txt* 290
- unicore* 290
- unmatch** 152, 161, 163
 - `.*` 165
 - atomic grouping 171
- unrolling the loop** 261-276
 - example 270-271, 477
 - general pattern 264
- `\p{Uppercase_Letter}` 123
- URL encoding 320
- URL example 74-77, 201-204, 208, 260, 303-304, 306, 320, 450-451
 - egrep* 25
 - Java 209
 - .NET 204
 - plucking 206-208
- use `charnames` 290
- use `Config` 290, 299
- use `English` 357
- use overload 342
 - (see also regex overloading)
- use re `'debug'` 361, 363
- use re `'eval'` 337
- use `strict` 295, 336, 345
- use `Time::HiRes` 358, 360
- use `warnings` 326, 363
- useAnchoringBounds method** 388
- usePattern method** 393, 399
- username example** 73, 76, 98
 - plucking from text 71-73
 - in URL 74-77
- useTransparentBounds method** 387
- using `System.Text.RegularExpressions` 416
- UTF-16 encoding 107
- UTF-8 encoding 107, 442, 447
- `\v` 115-116, 364
- `\V` 364
- Value
 - Group object method 430
 - Match object method 427
- variable names example** 24
- variables
 - after match
 - pre-match copy 355
 - binding 339
 - fully qualified 295
 - interpolation 344
 - naughty 356
 - tied 299
- VB.NET xvii
 - code example 204, 219
 - comments 99
 - regex approach 96-97
 - strings 103
 - (see also .NET)
- verbatim strings** 103
- Version 7 regex** 183
- Version 8 regex** 183
- version covered
 - Java 365
 - .NET 405
 - Perl 283
 - PHP 440
 - others* 91
- version history** Java 365, 368-369, 392, 401
- vertical tab** 109, 370
 - Perl `\s` 288
- vi* after-match data 138
- Vietnamese text processing** 29
- virtual machine** 236
- Visual Basic xvii
 - (see also VB.NET)
 - (see also .NET)
- Visual Studio .NET 434
- VM 236
 - Java 236
 - warming up 236

- void context** 294
- VT** 109, 370
-
- \$^w** 297
- \w** 49, 65, 120
 - Emacs 129
 - Java 368
 - many different interpretations 93
 - Perl 288
 - PHP 120, 442
- \W** 49, 121
- Wall, Larry** 88-90, 140, 363
- warning up Java VM** 236
- warnings** 296
 - (\$^w variable)
 - Perl 297
 - Perl 38
 - temporarily turning off 297
 - use warnings
 - Perl 326, 363
- warnings pragma** 326, 363
- while vs. foreach vs. if** 320
- whitespace**
 - allowing optional 18
 - removing 199-200
- width attribute** Java example 397
- wildcards** filename 4
- word anchor** mechanics of matching 150
- word boundaries** 133
 - \<-\>
 - egrep* 15
 - introduced 15
 - Java 134
 - many programs 134
 - mimicking 66, 134, 341-342
 - .NET 134
 - Perl 288
 - PHP 134
- www.cpan.org** 358
- www.PeakWebbosting.com** xxiv
- www.regex.info** 358
- www.unixwiz.net** xxiv, 458
-
- \x** 108, 120
- /x** 135, 288
 - (see also: comments and free-spacing mode; mode modifier)
 - history 90
 - introduced 72
- (?x)** (see: comments and free-spacing mode; mode modifier)
-
- \x** 117, 406
 - Perl 286
- XML** 483
 - CDATA 483
- XML example** 481-484
-
- y** old *grep* 86
- ¥** 124
- Yahoo!** xxiv, 74, 132, 190, 206-207, 258, 314, 397
-
- \z** 112, 129-130
 - (see also enhanced line-anchor mode)
 - Java 370
 - optimization 246
- \p{z}** 121-122, 368, 407
- \z** 112, 129-130, 316, 447
 - (see also enhanced line-anchor mode)
 - optimization 246
 - PHP 442
- Zawodny, Jeremy** 258
- zero-width assertions** (see: anchor; look-ahead; lookbehind)
- ZIP code example** 209-212
- \p{z1}** 123
- Zmievski, Andrei** xxiv, 440
- \p{zp}** 123
- \p{zs}** 123