
Table of Contents

<i>Preface</i>	<i>xv</i>
<i>1: Introduction to Regular Expressions</i>	<i>1</i>
Solving Real Problems	2
Regular Expressions as a Language	4
The Filename Analogy	4
The Language Analogy	5
The Regular-Expression Frame of Mind	6
If You Have Some Regular-Expression Experience	6
Searching Text Files: Egrep	6
Egrep Metacharacters	8
Start and End of the Line	8
Character Classes	9
Matching Any Character with Dot	11
Alternation	13
Ignoring Differences in Capitalization	14
Word Boundaries	15
In a Nutshell	16
Optional Items	17
Other Quantifiers: Repetition	18
Parentheses and Backreferences	20
The Great Escape	22
Expanding the Foundation	23
Linguistic Diversification	23
The Goal of a Regular Expression	23

A Few More Examples	23
Regular Expression Nomenclature	27
Improving on the Status Quo	30
Summary	32
Personal Glimpses	33
2: <i>Extended Introductory Examples</i>	35
About the Examples	36
A Short Introduction to Perl	37
Matching Text with Regular Expressions	38
Toward a More Real-World Example	40
Side Effects of a Successful Match	40
Intertwined Regular Expressions	43
Intermission	49
Modifying Text with Regular Expressions	50
Example: Form Letter	50
Example: Prettifying a Stock Price	51
Automated Editing	53
A Small Mail Utility	53
Adding Commas to a Number with Lookaround	59
Text-to-HTML Conversion	67
That Doubled-Word Thing	77
3: <i>Overview of Regular Expression Features and Flavors</i>	83
A Casual Stroll Across the Regex Landscape	85
The Origins of Regular Expressions	85
At a Glance	91
Care and Handling of Regular Expressions	93
Integrated Handling	94
Procedural and Object-Oriented Handling	95
A Search-and-Replace Example	97
Search and Replace in Other Languages	99
Care and Handling: Summary	101
Strings, Character Encodings, and Modes	101
Strings as Regular Expressions	101
Character-Encoding Issues	105
Regex Modes and Match Modes	109
Common Metacharacters and Features	112
Character Representations	114

Character Classes and Class-Like Constructs	117
Anchors and Other “Zero-Width Assertions”	127
Comments and Mode Modifiers	133
Grouping, Capturing, Conditionals, and Control	135
Guide to the Advanced Chapters	141
4: <i>The Mechanics of Expression Processing</i>	143
Start Your Engines!	143
Two Kinds of Engines	144
New Standards	144
Regex Engine Types	145
From the Department of Redundancy Department	146
Testing the Engine Type	146
Match Basics	147
About the Examples	147
Rule 1: The Match That Begins Earliest Wins	148
Engine Pieces and Parts	149
Rule 2: The Standard Quantifiers Are Greedy	151
Regex-Directed Versus Text-Directed	153
NFA Engine: Regex-Directed	153
DFA Engine: Text-Directed	155
First Thoughts: NFA and DFA in Comparison	156
Backtracking	157
A Really Crummy Analogy	158
Two Important Points on Backtracking	159
Saved States	159
Backtracking and Greediness	162
More About Greediness and Backtracking	163
Problems of Greediness	164
Multi-Character “Quotes”	165
Using Lazy Quantifiers	166
Greediness and Laziness Always Favor a Match	167
The Essence of Greediness, Laziness, and Backtracking	168
Possessive Quantifiers and Atomic Grouping	169
Possessive Quantifiers, ?+, *+, ++, and {m,n}+	172
The Backtracking of Lookaround	173
Is Alternation Greedy?	174
Taking Advantage of Ordered Alternation	175
NFA, DFA, and POSIX	177

“The Longest-Leftmost”	177
POSIX and the Longest-Leftmost Rule	178
Speed and Efficiency	179
Summary: NFA and DFA in Comparison	180
Summary	183
5: Practical Regex Techniques	185
Regex Balancing Act	186
A Few Short Examples	186
Continuing with Continuation Lines	186
Matching an IP Address	187
Working with Filenames	190
Matching Balanced Sets of Parentheses	193
Watching Out for Unwanted Matches	194
Matching Delimited Text	196
Knowing Your Data and Making Assumptions	198
Stripping Leading and Trailing Whitespace	199
HTML-Related Examples	200
Matching an HTML Tag	200
Matching an HTML Link	201
Examining an HTTP URL	203
Validating a Hostname	203
Plucking Out a URL in the Real World	205
Extended Examples	208
Keeping in Sync with Your Data	208
Parsing CSV Files	212
6: Crafting an Efficient Expression	221
A Sobering Example	222
A Simple Change—Placing Your Best Foot Forward	223
Efficiency Verses Correctness	223
Advancing Further—Localizing the Greediness	225
Reality Check	226
A Global View of Backtracking	228
More Work for a POSIX NFA	229
Work Required During a Non-Match	230
Being More Specific	231
Alternation Can Be Expensive	231
Benchmarking	232

Know What You're Measuring	234
Benchmarking with Java	234
Benchmarking with VB.NET	236
Benchmarking with Python	237
Benchmarking with Ruby	238
Benchmarking with Tcl	239
Common Optimizations	239
No Free Lunch	240
Everyone's Lunch is Different	240
The Mechanics of Regex Application	241
Pre-Application Optimizations	242
Optimizations with the Transmission	245
Optimizations of the Regex Itself	247
Techniques for Faster Expressions	252
Common Sense Techniques	254
Expose Literal Text	255
Expose Anchors	255
Lazy Versus Greedy: Be Specific	256
Split Into Multiple Regular Expressions	257
Mimic Initial-Character Discrimination	258
Use Atomic Grouping and Possessive Quantifiers	259
Lead the Engine to a Match	260
Unrolling the Loop	261
Method 1: Building a Regex From Past Experiences	262
The Real "Unrolling-the-Loop" Pattern	263
Method 2: A Top-Down View	266
Method 3: An Internet Hostname	267
Observations	268
Using Atomic Grouping and Possessive Quantifiers	268
Short Unrolling Examples	270
Unrolling C Comments	272
The Freeflowing Regex	277
A Helping Hand to Guide the Match	277
A Well-Guided Regex is a Fast Regex	279
Wrapup	280
In Summary: Think!	281

7: Perl	283
Regular Expressions as a Language Component	285
Perl's Greatest Strength	286
Perl's Greatest Weakness	286
Perl's Regex Flavor	286
Regex Operands and Regex Literals	288
How Regex Literals Are Parsed	292
Regex Modifiers	292
Regex-Related Perlisms	293
Expression Context	294
Dynamic Scope and Regex Match Effects	295
Special Variables Modified by a Match	299
The <code>qr/--/</code> Operator and Regex Objects	303
Building and Using Regex Objects	303
Viewing Regex Objects	305
Using Regex Objects for Efficiency	306
The Match Operator	306
Match's Regex Operand	307
Specifying the Match Target Operand	308
Different Uses of the Match Operator	309
Iterative Matching: Scalar Context, with <code>/g</code>	312
The Match Operator's Environmental Relations	316
The Substitution Operator	318
The Replacement Operand	319
The <code>/e</code> Modifier	319
Context and Return Value	321
The Split Operator	321
Basic Split	322
Returning Empty Elements	324
Split's Special Regex Operands	325
Split's Match Operand with Capturing Parentheses	326
Fun with Perl Enhancements	326
Using a Dynamic Regex to Match Nested Pairs	328
Using the Embedded-Code Construct	331
Using <code>local</code> in an Embedded-Code Construct	335
A Warning About Embedded Code and <code>my</code> Variables	338
Matching Nested Constructs with Embedded Code	340
Overloading Regex Literals	341
Problems with Regex-Literal Overloading	344

Mimicking Named Capture	344
Perl Efficiency Issues	347
“There’s More Than One Way to Do It”	348
Regex Compilation, the /o Modifier, <code>qr/--/</code> , and Efficiency	348
Understanding the “Pre-Match” Copy	355
The Study Function	359
Benchmarking	360
Regex Debugging Information	361
Final Comments	363
8: Java	365
Judging a Regex Package	366
Technical Issues	366
Social and Political Issues	367
Object Models	368
A Few Abstract Object Models	368
Growing Complexity	372
Packages, Packages, Packages	372
Why So Many “Perl5” Flavors?	375
Lies, Damn Lies, and Benchmarks	375
Recommendations	377
Sun’s Regex Package	378
Regex Flavor	378
Using <code>java.util.regex</code>	381
The <code>Pattern.compile()</code> Factory	383
The <code>Matcher</code> Object	384
Other <code>Pattern</code> Methods	390
A Quick Look at Jakarta-ORO	392
ORO’s <code>Perl5Util</code>	392
A Mini <code>Perl5Util</code> Reference	393
Using ORO’s Underlying Classes	397
9: .NET	399
.NET’s Regex Flavor	400
Additional Comments on the Flavor	402
Using .NET Regular Expressions	407
Regex Quickstart	407
Package Overview	409
Core Object Overview	410

Core Object Details	412
Creating <code>Regex</code> Objects	413
Using <code>Regex</code> Objects	415
Using <code>Match</code> Objects	421
Using <code>Group</code> Objects	424
Static “Convenience” Functions	425
Regex Caching	426
Support Functions	426
Advanced .NET	427
Regex Assemblies	428
Matching Nested Constructs	430
Capture Objects	431
<i>Index</i>	433