Mastering Perl

by brian d foy
Stonehenge Consulting Services, Inc.

version 1.6
February 2, 2009

Table of Contents

Introduction

About this course Sec1:2
The path to mastery Sec1:3

Modulinos

Programs versus modules 5
Bring back main() 6
Tell Perl where to start 7
Make it a module 8
Who’s calling? 9
caller() in a module 10
Compile as a module, run as a program 11
Testing our program 12
Adding to the program 13
Packaging 15
Wrapper programs 16
Installing programs 17
Other methods 18
Distribute through CPAN 19
Conclusion 20
Further reading 21

Configuration

Configuration goals 23
Configuration techniques 24
The wrong way 25
Slightly better (still bad) 26
Environment variables 27
Set defaults 28
Perl’s Config 29
Command-line switches 30
perl’s -s switch 31
Getopt::Std and getopt 32
Getopt::Std and getopts 33
Getopt::Long 34
More GetOpt::Long 35
Extreme and odd cases 36
Configuration files 37
ConfigReader::Simple 38
INI Files 39
Config::IniFiles 40
Config::Scoped 41
AppConfig 42
Using the program name 43
By operating system 44
Writing your own interface 45
Good method names 46
Further reading 47

Lightweight Persistence

Persistence 49

Perl structures as text 50
Using my own name 51
Nicer output 52
Reading Data::Dumper text 53
YAML Ain’t Markup 54
YAML format 55
Reading in YAML 56
Storable 57
Reading Storable files 58
Freezing and thawing 59
Storing multiple values 60
Deep copies 61
dbm files (old, trusty) 62
A better DBM 63
Further reading 64

Dynamic Subroutines

Just what is “dynamic”? 66
You’re soaking in it! 67
A typical dispatch table 68
A review of subroutine references 69
Subroutines as data 70
Add additional operators 71
Create pipelines 72
Validate data with pipelines 73
Store the validation profile as text 74
Serialize my code 75
Replace named subroutines 76
Redefine subs in other packages 77
Export subroutines 78
Create new subs with AUTOLOAD 79
Mock subroutines 80
Fixing modules 81
Wrapping subroutines 82
Subroutines as arguments 83
Summary 84
Further reading 85

Logging

Log without changing the program 87
Two major modules 88
The :easy way 89
Logging levels 90
Something more complex 91
Configuring Log4perl 92
Appenders handle the magic 93
Logging to a database 94
Changing configuration on-the-fly 95
Send to screen and file at once 96
Multiple loggers 97
Further reading 98

Profiling

Profiling is better than benchmarking 100
A recursive subroutine 101
Calling a Profiler 102

Recursion profile 103
Iteration, not recursion 104
Iteration profile 105
Really big numbers 106
Memoize 107
What happened? 108
More complex profiling 109
Modern profiling with NYTProf 110
The basics of profiling 111
Record DBI queries 112
Database optimization 113
Profiling DBI Statements 114
Profiling DBI methods 115
Profiling test suites 116
Devel::Cover HTML report 117
Devel::Cover detail 118
Further reading 119

Benchmarking

Measuring Perl 121
Theory of measurement 122
Know where you are 123
Using benchmarks 124
Single points 125
Multiple points 126
All things being equal 127
Don’t benchmark languages 128
Definitions of performance 129
Possible metrics 130
Devel::Peek 131
Memory use 132
About Benchmark.pm 133
Time a single bit of code 134
Compare several bits of code 135
Common misuse 136
Do these numbers make sense? 137
Report the situation 138
Do something useful 139
Now the results make sense 140
Verify with an experiment 141
Benchmarking summary 142
Further reading 143

Conclusion

Main points 145
More information 146

Questions

Mastering Perl

by brian d foy
Stonehenge Consulting Services, Inc.
version 1.6
February 2, 2009

Introduction

About this course

- Selected topics for the working programmer based on *Mastering Perl*
- Mostly not about syntax or wizardly tricks
- Not for masters, but people who want to control Perl code
- Not necessarily the way to do it, just the way I've done it
- Create "professional", robust programs other people can use
- We'll cover
  - profiling
  - benchmarking
  - configuration
  - logging
  - lightweight persistence

The path to mastery

- The guild system had a progression of skills
- Apprentices were the beginners and worked with supervision
- Journeymen were competent in their trade
- Masters taught journeymen
- Journeymen studied under different masters
    - different masters teach different tricks and methods
    - journeyman develop their own style
- A masterpiece showed that a journeyman mastered his trade