

Perl tutorial

The Perl Maven tutorial will teach you the basics of the Perl programming language. You'll be able to write simple scripts, analyze log files and read and write CSV files. Just to name a few common tasks.

[Next](#)

You'll learn how to use the CPAN and several specific CPAN modules.

It will be a good foundation for you to build on.

The free on-line version of the tutorial is currently in development. Many parts are ready. Additional parts are being published every few days. The latest one was published on May 12, 2016. If you are interested in getting updated when new parts are published, please [subscribe to the newsletter](#).

There is also an [e-book](#) version of the material available for purchase. In addition to the free tutorial, that version also includes the slides from the corresponding course including many exercises and their solutions. The course material covers all the parts, including the areas that are not yet covered in the free version.

The companion [video-course](#) includes over 210 screencasts, a total of more than 5 hours of video. In addition to presenting the material it also provides explanations to the solutions of all the exercise. The package also includes the source code of all the examples and exercises.

Free on-line Beginner Perl Maven tutorial

In this tutorial you are going to learn how to use the Perl 5 programming language to **get your job done**.

You will learn both general language features, and extensions or libraries or as the Perl programmers call them **modules**. We will see both standard modules, that come with perl and 3rd-party modules, that we install from **CPAN**.

When it is possible I'll try to teach things in a very task oriented way. I'll draw up tasks and then we'll learn the necessary tools to solve them. Where possible I'll also direct you to some exercises you can do to practice what you have learned.

Introduction

1. [Install Perl, print Hello World, Safety net \(use strict, use warnings\)](#)
2. [#!/usr/bin/perl - the hash-bang line](#)
3. [Editors, IDEs, development environment for Perl](#)
4. [Getting Help](#)
5. [Perl on the command line](#)
6. [Core Perl documentation, CPAN module documentation](#)
7. [POD - Plain Old Documentation](#)
8. [Debugging Perl scripts](#)

Scalars

1. Common warnings and error messages
2. Prompt, read from STDIN, read from the keyboard
3. Automatic string to number conversion
4. Conditional statements: if
5. Boolean (true and false) values in Perl
6. Numerical operators
7. String operators
8. undef, the initial value and the defined function
9. Strings in Perl: quoted, interpolated and escaped
10. Here documents
11. Scalar variables
12. Comparing scalars
13. String functions: length, lc, uc, index, substr
14. Number Guessing game (rand, int)
15. Perl while loop
16. Scope of variables in Perl
17. Boolean Short circuit

Files

1. exit
2. Standard Output, Standard Error and command line redirection
3. warn
4. die
5. Writing to files
6. Appending to files
7. Open and read from files using Perl
8. Don't open files in the old way
9. Slurp mode

Lists and Arrays

1. The for loop in Perl
2. Arrays in Perl
3. Process command line parameters @ARGV
4. Process command line parameters using Getopt::Long
5. Advanced usage of Getopt::Long for accepting command line arguments
6. split
7. How to read and process a CSV file? (split, Text::CSV_XS)
8. join
9. The year of 19100 (time, localtime, gmtime) and introducing context
10. Context sensitivity in Perl
11. Reading from a file in scalar and list context
12. STDIN in scalar and list context
13. Sorting arrays in Perl
14. Sorting mixed strings
15. Unique values in an array in Perl
16. Manipulating Perl arrays: shift, unshift, push, pop
17. Stack and queue
18. reverse
19. The ternary operator
20. qw - quote word

Subroutines

1. [Subroutines and Functions in Perl](#)
2. [Variable number of parameters](#)
3. [Recursive subroutines](#)

Hashes, arrays

1. [Perl Hashes \(dictionary, associative array, look-up table\)](#)
2. [Creating hash from an array](#)
3. [Perl hash in scalar and list context](#)
4. [Sorting a hash](#)
5. [Count word frequency in a text file](#)

Regular Expressions

1. [Introduction to Regular Expressions in Perl](#)
2. [Regex: character classes](#)
3. [Regex: special character classes](#)
4. [Regex: quantifiers](#)
5. [Regex videos - part I](#)
6. [trim - remove leading and trailing spaces](#)

Perl and Shell related functionality

1. [How to remove, copy or rename a file with Perl](#)
2. [Directory handles](#)
3. [Traversing directory tree manually with recursion, manually using a queue and using find.](#)

CPAN

1. [Download and install Perl \(Strawberry Perl or manual compilation\)](#)
2. [How to change @INC to find Perl modules in non-standard locations?](#)
3. [How to change @INC to a relative directory](#)

Few examples for using Perl

1. [How to replace a string in a file with Perl? \(slurp\)](#)
2. [Reading Excel files using Perl](#)
3. [Creating Excel files using Perl](#)
4. [Sending e-mail using Perl](#)
5. [CGI scripts with Perl](#)
6. [Reading and writing JSON files](#)
7. [Database access using Perl \(DBI, DBD::SQLite, MySQL, PostgreSQL, ODBC\)](#)
8. [Accessing LDAP using Perl](#)

Common warnings and error messages

1. [Global symbol requires explicit package name also explained in Variable declaration in Perl](#)
2. [Use of uninitialized value](#)
3. [Bareword not allowed while "strict subs" in use](#)
4. [Name "main::x" used only once: possible typo at ...](#)
5. [Unknown warnings category](#)

6. Can't use string ("Foo") as a HASH ref while "strict refs" in use at ... explained in [Symbolic references in Perl](#)
7. [Can't locate ... in @INC](#)
8. [Scalar found where operator expected](#)
9. ["my" variable masks earlier declaration in same scope](#)
10. [Can't call method ... on unblessed reference](#)
11. [Argument ... isn't numeric in numeric ...](#)
12. [Can't locate object method "..." via package "1" \(perhaps you forgot to load "1"?\)](#)
13. [Odd number of elements in hash assignment](#)
14. [Possible attempt to separate words with commas](#)

Other

1. [Splice to slice and dice arrays in Perl](#)
2. [Multi dimensional arrays](#)
3. [Multi dimensional hashes](#)
4. [Minimal requirement to build a sane CPAN package](#)
5. [What are string and numeric contexts?](#)
6. [Statement modifiers: reversed if statements](#)

Object Oriented Perl with Moose or Moo

There is a whole series of articles on writing Object Oriented code, using the light-weight [Moo](#) OOP framework or the full-blown [Moose](#)OOP framework.

[How to improve your Perl code](#)

Just a reminder, there are corresponding [e-books](#) and [video courses](#)available.