Learn Security Online

Perl Reference Sheet

This document provides a brief reference to information that is commonly needed by a perl programmer.

Command Line options

This is a summary of commonly used command line options
- -c   Checks syntax but does not execute
- -d   Runs the script under the debugger
- -D   NUMBER sets debugging flag
- -e   Used to execute script on the command line
- -T   Forces taint checking
- -w   Print warning

Variables

$var is a scalar variable
@array is an array
%var is a hash

Literals

Numeric	String
123	'abc' literal string no variable parsing
1_234	"abc" variable are parsed
123.4	`cmd` stores the output of ommand
2E-10	
0x01	
0337	

How to assign values

$var = 1; $var = "abc";

Arrays

(1, 2, 3)
(1..4) is the same as (1, 2, 3 ,4)
qw/value1 value2/ is the same as ("value1", "value2")
Arrays are referenced using [i] where i is the array element number.

How to assign values

```
@array = {"a", "c", "c"};
```

Assigning a value to a hash

```
$var->{'key'} = "value";
$$var{'key'} = "value";
```

Operators

Mathematical Operators

**	Exponentiation
+	Addition
-	Subtraction
*	Multiplication
/	Division
++	Increment by one
--	Decrement by one

Logical Operators

||	OR
&&	AND
==	Equal to for numeric comparisons
!=	Not equal to for numeric comparison
eq	Equal to for string comparisons
ne	Not equal to for string comparisons
<	Less than for numeric comparisons
>	Greater than for numeric comparisons
lt	Less than for string comparisons
gr	Greater than for string comparisons
<=>	Compare numeric value returns –1, 0 or 1
cmp	Compare string value returns –1, 0, or 1
=~	Search pattern
!~	Does not contain search pattern

Bit Operations

&	AND
|	OR
^	XOR
<<	Left shift
>>	Right shift

Other Operators

->	Reference hash element
.	Concatenate string values

Keywords

If statement

```
if      {        }
elsif   {        }
else    {        }
```

Loops

```
for (initialisation;comparison;perform each itteration)
do {      } while (expression);
do {      };
while (expression) {       };
```