An Overview of Perl

A language for Systems and Network Administration and Management:

An overview of the language

Where do I get Perl?

For Windows, go to http://www.activestate.com, download the installer

For Linux: it will be already installed

For other platforms: go to http://www.perl.com

This is a good source of other information about Perl

Where do I get Info about Perl?

Web sites:

http://www.perl.com

http://www.activestate.com

http://use.perl.org

On your hard disk:

perldoc –f *function*

Will look up the documentation for the built-in *function* (from the documentation perlfunc)

perldoc –q *word*

Will look up *word* in the headings of the FAQ

perldoc perl

A list of much of your locally installed documentation, divided into topics

ActiveState Perl provides a Programs menu item that links to online html documentation

CPAN, PPM: Many Modules

A very strong feature of Perl is the community that supports it

There are tens of thousands of third party modules for many, many purposes:

Eg. Net::LDAP module supports all LDAP operations, Net::LWP provides a comprehensive web client

Installation is easy:

sudo perl –MCPAN –e shell

install Net::LDAP

Will check if a newer version is available on the Internet from CPAN, and if so, download it, compile it, test it, and if it passes tests, install it.

PPM: Perl Package Manager

For Windows

Avoids need for a C compiler, other development tools

Download precompiled modules from ActiveState and other sites, and install them:

ppm install Net::LDAP

See documentation with ActiveState Perl

Mailing Lists: help from experts

There are many mailing lists and newsgroups for Perl

When subscribe to mailing list, receive all mail from list

When send mail to list, all subscribers receive

For Windows, many lists at http://www.activestate.com

How to ask Questions on a List

I receive many email questions from students about many topics

Most questions are not clear enough to be able to answer in any way except, "please tell me more about your problem"

Such questions sent to mailing lists are often unanswered

Need to be concise, accurate, and clear

see also Eric Raymond's *How to Ask Questions the Smart Way* at
http://catb.org/~esr/faqs/smart-questions.html

Search the FAQs first

Where is Perl on my system?

ActiveState Perl installs **perl.exe** in **C:\Perl\perl.exe**

Linux systems have a standard location for perl at
/usr/bin/perl

On some UNIX systems, it may be installed at
/usr/local/bin/perl

How does my OS know it's a Perl program?

To run your Perl program, OS needs to call perl

How does OS know when to call Perl?

Windows:

OS uses the extension of the file to decide what to do (e.g., **.bat**, **.exe**)

Your program names end with **.pl**

Linux, Unix:

programs have *execute* permission (**chmod +x *program***)

OS reads first 2 bytes of program: if they are "**#!**" then read to end of line, then use that as the interpreter

OS doesn't care what your program file is called

For cross platform support:

Put this at the top of all your programs:

#! /usr/bin/perl –w

Name your programs with an extension **.pl**

Language Overview

A quick look at more important features of the language

Funny Characters $, @, %

Variables in Perl start with a *funny character*

Why?

No problem with reserved words:

can have a variable called $while, and another variable called @while, and a third called %while.

Can *interpolate* value into a string:

my $string = "long";

my $number = 42.42;

print "my string is $string and my number is $number\n";

Arrays

Define an array like this:

my @array = (1, 5, "fifteen");

This is an array containing three elements

The first can be accessed as $array[0], second as $array[1], the last as $array[2]

Note that since each element is a scalar, it has the $ funny character for a scalar variable *value*

In Perl, we seldom use an array with an index—use list processing array operations

higher level.

Hashes

Hashes are probably new to you

Like an array, but indexed by a string

Similar idea was implemented in java.lang.HashTable

Perl hashes are easier to use

Example:

my %hash = (NL => 'Netherlands',

BE => 'Belgium');

This creates a hash with two elements

first is $hash{NL}, has value "Netherlands";