DATABASE TECHNOLOGY

Perl
A DBA and Developers best (forgotten) friend

A beginners guide to Perl

www.dbvisit.com

Gold Partner

Introduction

Arjen Visser
Founder and CTO of Dbvisit Software Limited

Creators of Dbvisit Standby and Dbvisit Replicate

Past Experience:

- DBA / Technical Director
- Team leader/Unix admin/project manager
- Datawarehouse developer/programmer
- Speaker at OOW 2009, 2010, 2011, NZOUG, CLOUG, RMOUG11

Some of Dbvisit customers

Dbvisit
DATABASE TECHNOLOGY

MTV MUSIC TELEVISION®

NZX

Southern Cross Hospitals

Pythian
love your data

NZ TRANSPORT AGENCY
WAKA KOTAHI

MIGHTY RIVER POWER

Alcatel·Lucent

7 ELEVEn®

التجاري
Al-Tijari

ORION™ HEALTH

UN DP

fiserv.

RICOH

QBE

Reliance Industries Limited

OKLAHOMA
www.ok.gov

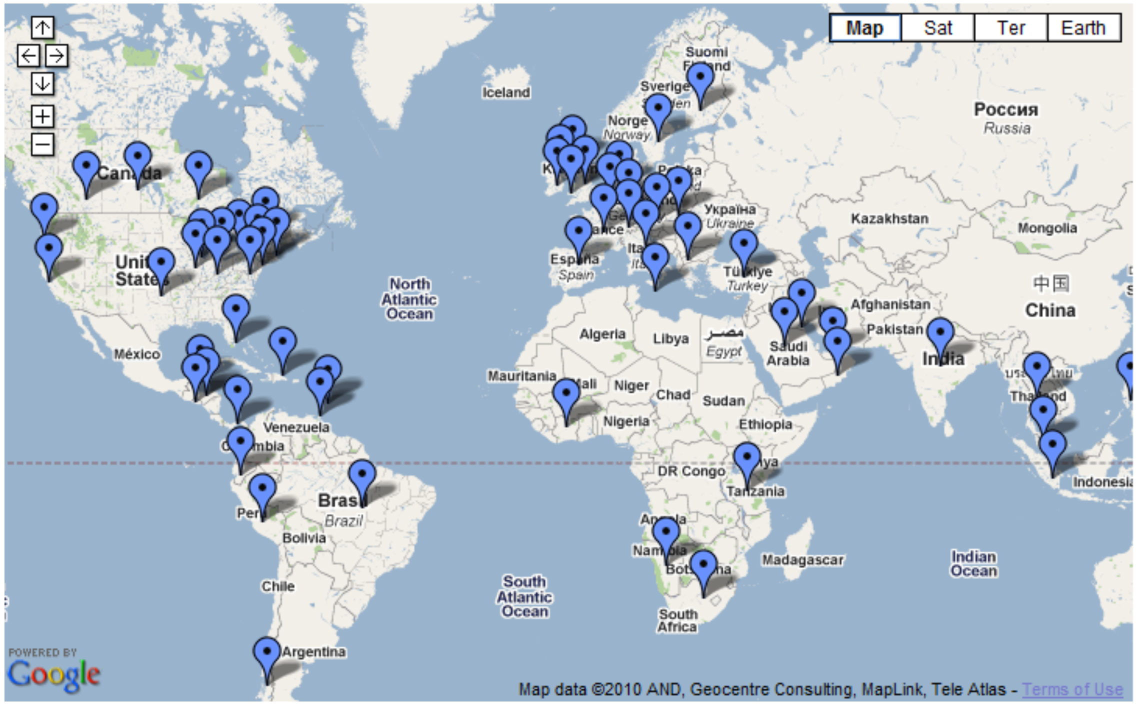
white pages

MANUKAU INSTITUTE OF TECHNOLOGY
Te Whare Takiura o Manukau

World #1 alternative to Data Guard

Dbvisit DATABASE TECHNOLOGY

Used by DBAs and companies over 60 countries



Dbvisit standby

Dbvisit replicate

Agenda

- What makes Perl so great
- What can I use Perl for / what not to use Perl for
- CPAN
- Brief language overview
- Making Perl portable
- Perl and Oracle
- Best way to learn Perl
- Small Perl project to take away

What is not covered

- Perl language in depth
- OO Perl
- Perl Comparison Python, Ruby etc

Brief history

One of the most portable languages around.

Larry Wall created Perl in 1987 while working at Unisys.

Motivation was because *awk* did not have the functionality he required.

Originally called Pearl.

Perl 5 was rewrite and released on October 17, 1994 and included:

- objects
- references
- modules

Current version is 5.14. (even numbers for production releases)

Future: Perl 6 will run on a cross-language virtual machine called Parrot.

Both Python and Perl will run on Parrot.

What makes Perl so great

Perception that is not as current as python, ruby, java etc…

Only a perception because:

- Oracle uses it in 11g.
  - asmcmd is written in Perl
  - Perl executable and libraries are standard installed with Oracle (also on Windows)
- VMware uses it.
- Dbvisit Standby and Replicate use it.
- Many websites use it (Amazon.com, bbc.co.uk, Zappos.com)
- CPAN (Comprehensive Perl Archive Network)
  - 15,000 modules by more than 7,000 authors - http://search.cpan.org/
  - DBD::Oracle to connect to Oracle
  - XML parsers
  - Encryption/security
  - email
  - Windows utilities (interface to register)
  - etc

What makes Perl so great (ii)

Advantages

- Interpreted language (not compiled)
- Concise language (write programs quickly)
- Allows handling of complex data structures
- You can get under the "hood" of Perl (like v$tables)
- Very strong pattern matching with regular expressions
- Easy to get started
- Tied in close to OS

Disadvantages

- Can be cryptic to read especially pattern matching
- No standard named parameters with calling functions (way around this)
- Hard to master (but is true for most languages)
- Not as fast as natively compiled programs such as C
- GUI applications written in Perl look dated (but you should not be writing GUI applications, should all be web based).

What can I use Perl for

Use Perl for

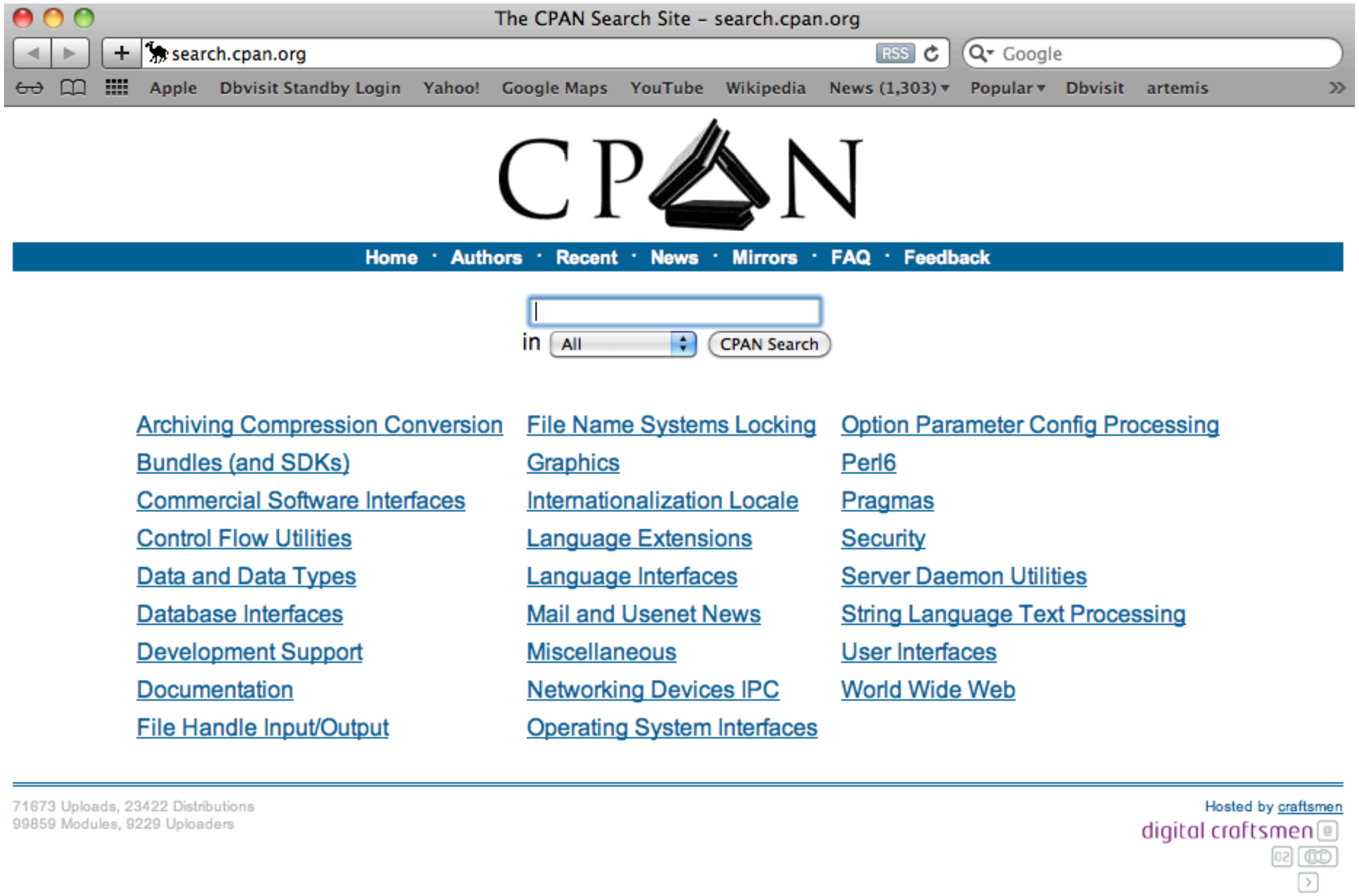
- Any shell or command line scripting or programs
- Batch type programming / Backend processing
- Data loading, manipulation (Data warehousing)
- Installation routines
- Heavy OS interfacing
- Web interface for batch processing

Maybe not use Perl (exclusively) for

- New Facebook/twitter web app
- Big corporate systems eg Billing/Financial
- Windows GUI application (like Thunderbird)

CPAN

Comprehensive Perl Archive Network


The CPAN Search Site – search.cpan.org

search.cpan.org

Apple Dbvisit Standby Login Yahoo! Google Maps YouTube Wikipedia News (1,303) Popular Dbvisit artemis

Home · Authors · Recent · News · Mirrors · FAQ · Feedback

in All CPAN Search

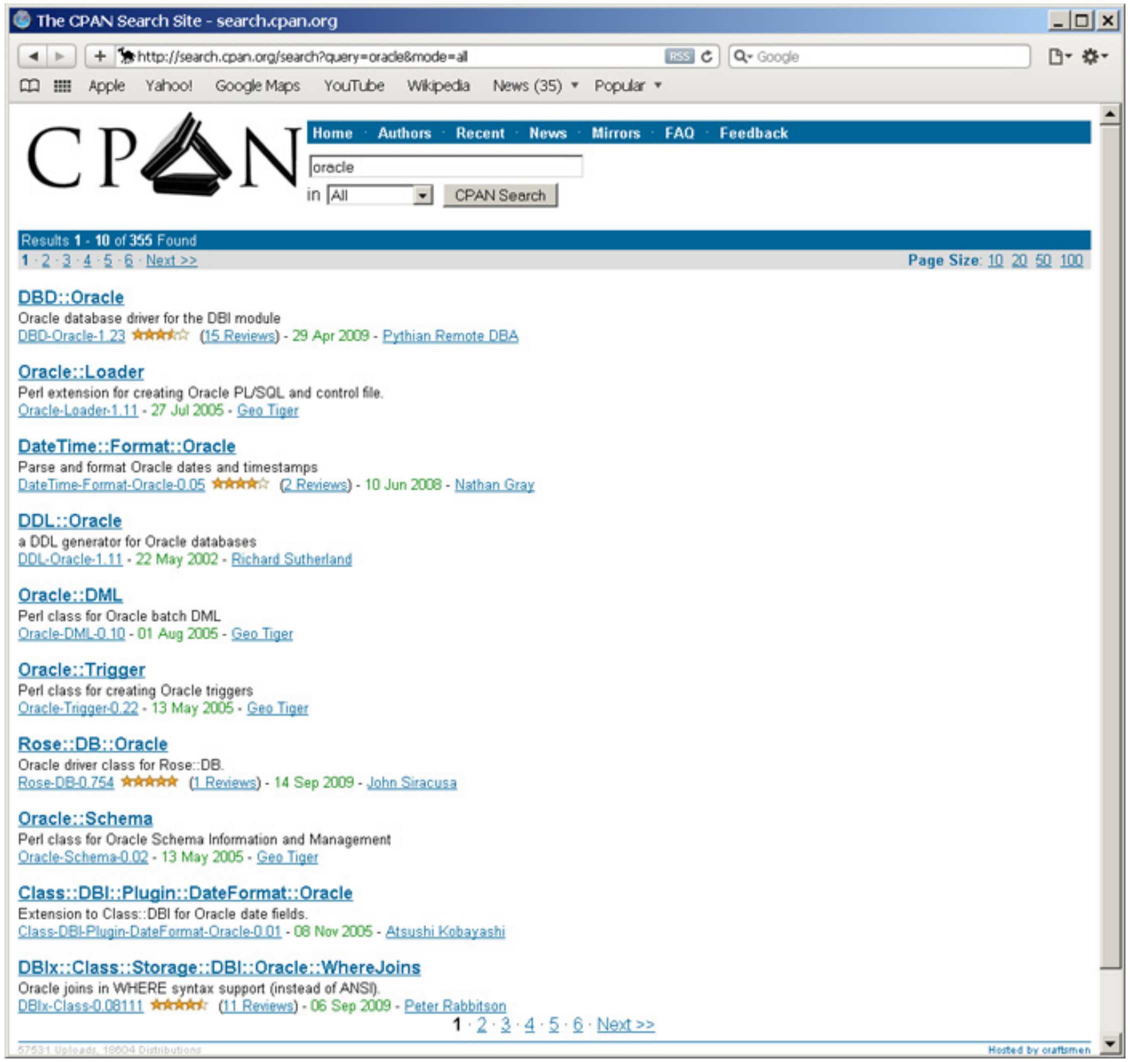
Archiving Compression Conversion	File Name Systems Locking	Option Parameter Config Processing
Bundles (and SDKs)	Graphics	Perl6
Commercial Software Interfaces	Internationalization Locale	Pragmas
Control Flow Utilities	Language Extensions	Security
Data and Data Types	Language Interfaces	Server Daemon Utilities
Database Interfaces	Mail and Usenet News	String Language Text Processing
Development Support	Miscellaneous	User Interfaces
Documentation	Networking Devices IPC	World Wide Web
File Handle Input/Output	Operating System Interfaces	

71673 Uploads, 23422 Distributions
99859 Modules, 9229 Uploaders

Hosted by craftsmen

digital craftsmen

CPAN


The CPAN Search Site - search.cpan.org

http://search.cpan.org/search?query=oracle&mode=all

Apple Yahoo! Google Maps YouTube Wikipedia News (35) Popular

Home · Authors · Recent · News · Mirrors · FAQ · Feedback

oracle

in All — CPAN Search

Results **1** - **10** of **355** Found

1 · 2 · 3 · 4 · 5 · 6 · Next >> — **Page Size:** 10 20 50 100

DBD::Oracle
Oracle database driver for the DBI module
DBD-Oracle-1.23 ★★★★☆ (15 Reviews) - 29 Apr 2009 - Pythian Remote DBA

Oracle::Loader
Perl extension for creating Oracle PL/SQL and control file.
Oracle-Loader-1.11 - 27 Jul 2005 - Geo Tiger

DateTime::Format::Oracle
Parse and format Oracle dates and timestamps
DateTime-Format-Oracle-0.05 ★★★★☆ (2 Reviews) - 10 Jun 2008 - Nathan Gray

DDL::Oracle
a DDL generator for Oracle databases
DDL-Oracle-1.11 - 22 May 2002 - Richard Sutherland

Oracle::DML
Perl class for Oracle batch DML
Oracle-DML-0.10 - 01 Aug 2005 - Geo Tiger

Oracle::Trigger
Perl class for creating Oracle triggers
Oracle-Trigger-0.22 - 13 May 2005 - Geo Tiger

Rose::DB::Oracle
Oracle driver class for Rose::DB.
Rose-DB-0.754 ★★★★★ (1 Reviews) - 14 Sep 2009 - John Siracusa

Oracle::Schema
Perl class for Oracle Schema Information and Management
Oracle-Schema-0.02 - 13 May 2005 - Geo Tiger

Class::DBI::Plugin::DateFormat::Oracle
Extension to Class::DBI for Oracle date fields.
Class-DBI-Plugin-DateFormat-Oracle-0.01 - 08 Nov 2005 - Atsushi Kobayashi

DBIx::Class::Storage::DBI::Oracle::WhereJoins
Oracle joins in WHERE syntax support (instead of ANSI).
DBIx-Class-0.08111 ★★★★☆ (11 Reviews) - 06 Sep 2009 - Peter Rabbitson

1 · 2 · 3 · 4 · 5 · 6 · Next >>

57531 Uploads, 18604 Distributions — Hosted by craftsmen

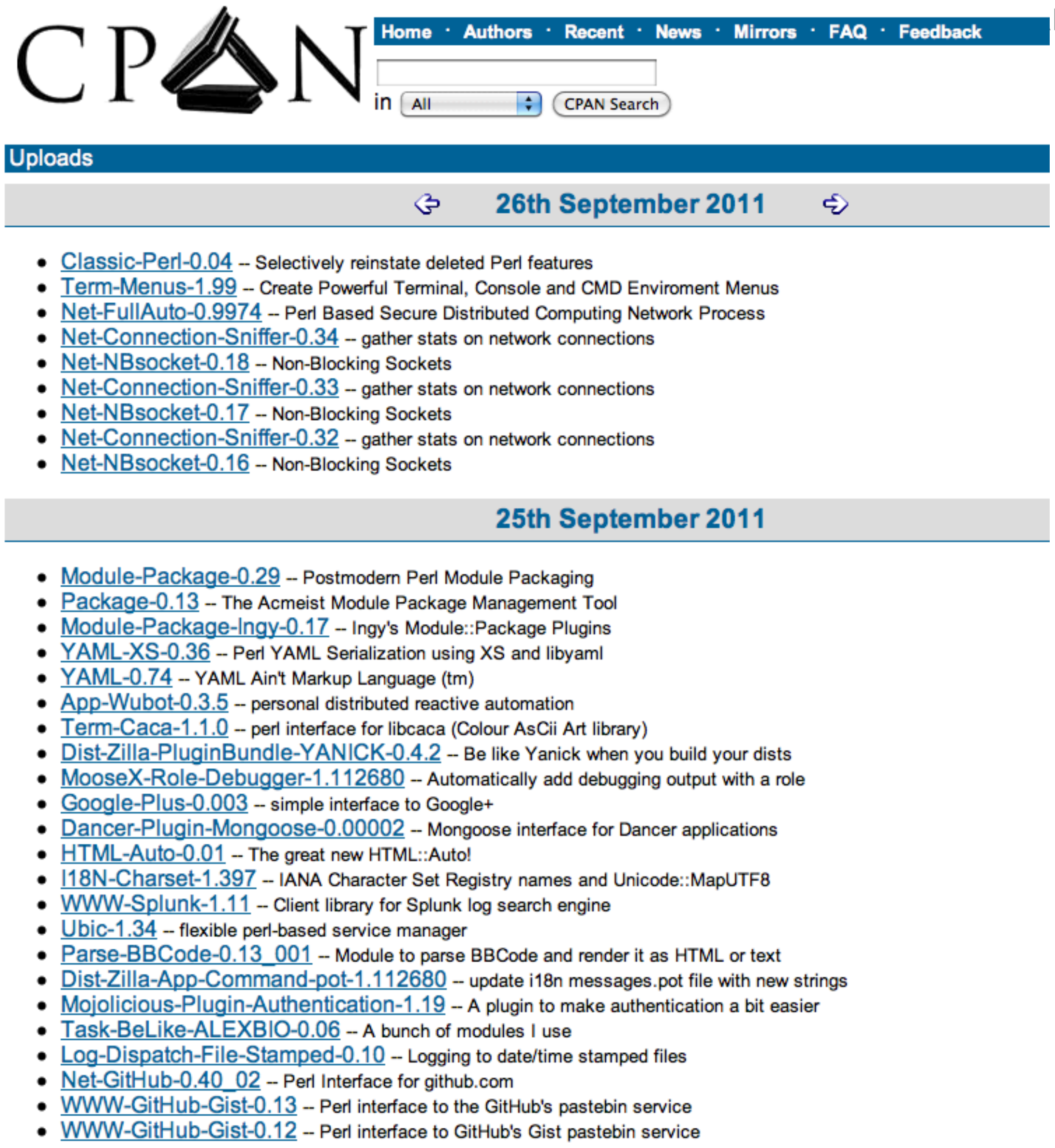
CPAN

Is CPAN Current?

Snapshot taken

26 Sept 2011

YES! ->



Uploads

26th September 2011

- Classic-Perl-0.04 -- Selectively reinstate deleted Perl features
- Term-Menus-1.99 -- Create Powerful Terminal, Console and CMD Enviroment Menus
- Net-FullAuto-0.9974 -- Perl Based Secure Distributed Computing Network Process
- Net-Connection-Sniffer-0.34 -- gather stats on network connections
- Net-NBsocket-0.18 -- Non-Blocking Sockets
- Net-Connection-Sniffer-0.33 -- gather stats on network connections
- Net-NBsocket-0.17 -- Non-Blocking Sockets
- Net-Connection-Sniffer-0.32 -- gather stats on network connections
- Net-NBsocket-0.16 -- Non-Blocking Sockets

25th September 2011

- Module-Package-0.29 -- Postmodern Perl Module Packaging
- Package-0.13 -- The Acmeist Module Package Management Tool
- Module-Package-Ingy-0.17 -- Ingy's Module::Package Plugins
- YAML-XS-0.36 -- Perl YAML Serialization using XS and libyaml
- YAML-0.74 -- YAML Ain't Markup Language (tm)
- App-Wubot-0.3.5 -- personal distributed reactive automation
- Term-Caca-1.1.0 -- perl interface for libcaca (Colour AsCii Art library)
- Dist-Zilla-PluginBundle-YANICK-0.4.2 -- Be like Yanick when you build your dists
- MooseX-Role-Debugger-1.112680 -- Automatically add debugging output with a role
- Google-Plus-0.003 -- simple interface to Google+
- Dancer-Plugin-Mongoose-0.00002 -- Mongoose interface for Dancer applications
- HTML-Auto-0.01 -- The great new HTML::Auto!
- I18N-Charset-1.397 -- IANA Character Set Registry names and Unicode::MapUTF8
- WWW-Splunk-1.11 -- Client library for Splunk log search engine
- Ubic-1.34 -- flexible perl-based service manager
- Parse-BBCode-0.13_001 -- Module to parse BBCode and render it as HTML or text
- Dist-Zilla-App-Command-pot-1.112680 -- update i18n messages.pot file with new strings
- Mojolicious-Plugin-Authentication-1.19 -- A plugin to make authentication a bit easier
- Task-BeLike-ALEXBIO-0.06 -- A bunch of modules I use
- Log-Dispatch-File-Stamped-0.10 -- Logging to date/time stamped files
- Net-GitHub-0.40_02 -- Perl Interface for github.com
- WWW-GitHub-Gist-0.13 -- Perl interface to the GitHub's pastebin service
- WWW-GitHub-Gist-0.12 -- Perl interface to GitHub's Gist pastebin service

How to use CPAN

On Unix/Linux with Internet connection:

```
$ cpan
cpan> i /google::pagerank/
Module    POE::Component::IRC::Plugin::Google::PageRank
Module    POE::Component::WWW::Google::PageRank
Module  = WWW::Google::PageRank
cpan> install WWW::Google::PageRank
```

On Unix/Linux without Internet connection:

Download zipped tar file from CPAN (`WWW-Google-PageRank-0.15.tar.gz`)

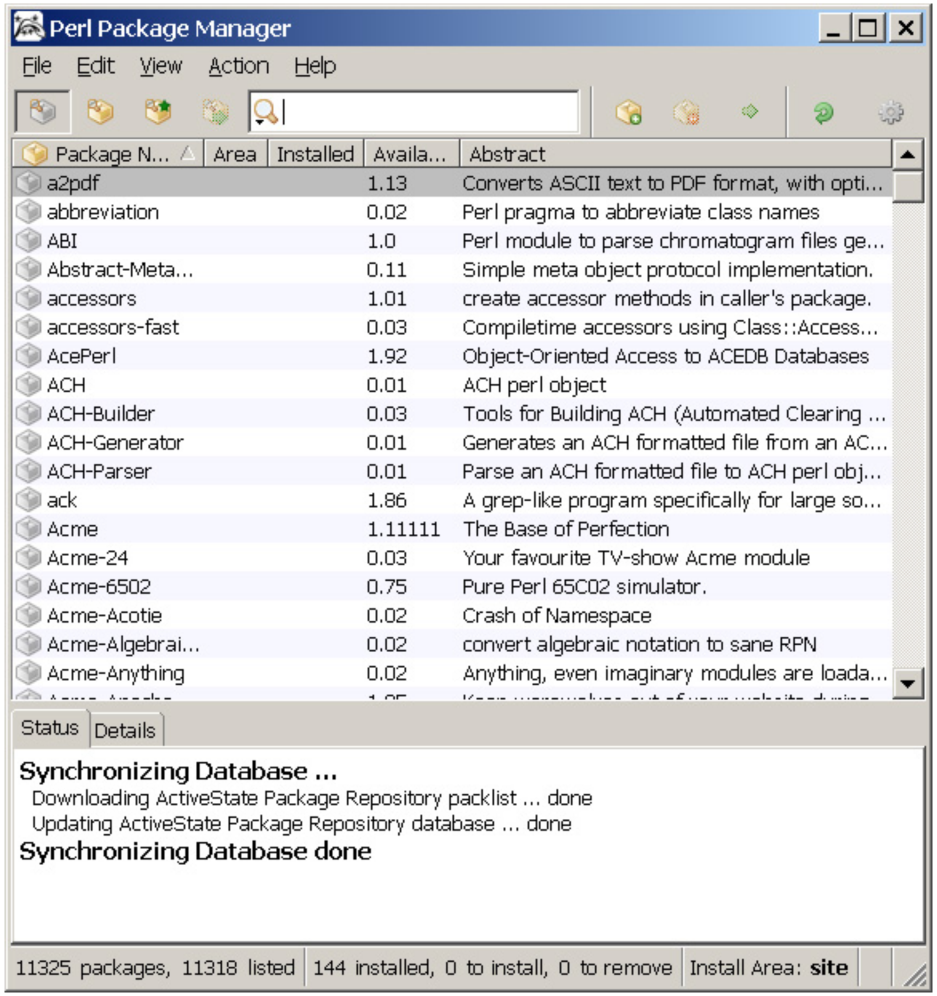
```
Unzip and untar file which will create a new temp directory.
cd into temp file directory:
perl Makefile.PL
make
make test
make install
Delete temp file directory.
```

How to use CPAN

On Windows

Require an Internet Connection

```
CMD$ ppm
```



Package N...	Area	Installed	Availa...	Abstract
a2pdf			1.13	Converts ASCII text to PDF format, with opti...
abbreviation			0.02	Perl pragma to abbreviate class names
ABI			1.0	Perl module to parse chromatogram files ge...
Abstract-Meta...			0.11	Simple meta object protocol implementation.
accessors			1.01	create accessor methods in caller's package.
accessors-fast			0.03	Compiletime accessors using Class::Access...
AcePerl			1.92	Object-Oriented Access to ACEDB Databases
ACH			0.01	ACH perl object
ACH-Builder			0.03	Tools for Building ACH (Automated Clearing ...
ACH-Generator			0.01	Generates an ACH formatted file from an AC...
ACH-Parser			0.01	Parse an ACH formatted file to ACH perl obj...
ack			1.86	A grep-like program specifically for large so...
Acme			1.11111	The Base of Perfection
Acme-24			0.03	Your favourite TV-show Acme module
Acme-6502			0.75	Pure Perl 65C02 simulator.
Acme-Acotie			0.02	Crash of Namespace
Acme-Algebrai...			0.02	convert algebraic notation to sane RPN
Acme-Anything			0.02	Anything, even imaginary modules are loada...

```
Synchronizing Database ...
  Downloading ActiveState Package Repository packlist ... done
  Updating ActiveState Package Repository database ... done
Synchronizing Database done
```

11325 packages, 11318 listed | 144 installed, 0 to install, 0 to remove | Install Area: **site**

Perl Package Manager is managed by ActiveState: http://www.activestate.com/

First look at Perl program!

Example or CPAN

```
# First complete Perl program!
use WWW::Google::PageRank;
#####
# Set website and get pagerank
#####
my $website = 'http://www.oracle.com';
my $pr = WWW::Google::PageRank->new();
my $rank = $pr->get($website);

#####
# Print out pagerank
#####
print "PageRank: " . $rank . " for website: $website";
```

Save program in file called pagerank.pl

```
perl pagerank.pl
PageRank: 8 for website: http://www.oracle.com
```

Even shorter, 3 lines!

```
use WWW::Google::PageRank;
my $pr = WWW::Google::PageRank->new;
print scalar($pr->get('http://www.oracle.com/')), "\n";
```

Save program in file called pagerank.pl

perl pagerank.pl

PageRank: 8 for website: http://www.oracle.com

Language overview

Perl brief language overview – 10 slides (20 minutes)

Including questions with prizes!

Brief language overview (slide 1 of 10)

Variable types:

▪	Scalar: `$`	(any single value)
▪	Array : `@`	(ordered list – has an index)
▪	Hash : `%`	(unordered set of scalars – key value pairs)

Scalar

```
$answer = 42;
$colour="red";
$cwd = `pwd`;
```

Array

```
@versions = ("7.3","8i","9i","10g","11g");
$versions[0] = "7.3";
$versions[1] = "8i";
@ordered_versions = sort @versions;
```

Hash

```
%summer = ( "dec" => "December", "jan" => "January", "feb" =>
"February");
```

Brief language overview (slide 2 of 10)

Declaring - local variables

```
my $i;
my @array_versions;
my $i = 2;
```

Declaring - global variables

```
our $Logfile;
our @Datafiles;
our %Ora_init;
```

Increment

```
$i++; # $i = $i + 1;
$i--; # $i = $i - 1;
```

Perl Question 1

Brief language overview (slide 3 of 10)

All lines end with ;

Comparisons

`==`	Numeric comparison. eg test if (`$pi == 3.14159`)
`>=`	Greater than
`<=`	Smaller than
`!=`	Not equal to
`eq`	String comparison (equal) (`$colour eq "red"`)
`ne`	String comparison (not equal)

Assignment

`=` Assignment (`$a = $b`) Remember this is not comparing

Brief language overview (slide 4 of 10)

Control structures:

```
while ( cond ) { ... }
for ( init-expr ; cond-expr ; incr-expr ) { ... }
foreach var ( list ) { ... }
if ( cond ) { ... }
if ( cond ) { ... } else { ... }
if ( cond ) { ... } elsif ( cond ) { ... } else { ... }

if ($version eq "8i" ) {
    print "Your Oracle version is old, but still good!\n"
}
if ($age == 40 ) {
    print "What a great age!\n"
}
```

(\n is newline)

Brief language overview (slide 5 of 10)

Internal build in variables

$_

(default internal variable when variable is not explicitly mentioned)

With $_

```
foreach (@colours) {
   print "colour is: $_\n";
}
```

With explicit variable

```
foreach $colour (@colours)
    print "colour is: $colour\n";
}
```

`$@`	Output of `eval` command
`$?`	Return code of child program
`$!`	OS error (example open file, or file delete error)
`$^O`	OS name (Solaris, Linux, Windows etc)
`@ARGV`	Array containing the arguments to the program

```
my ($db, $oracle_home) = @ARGV;
```

Brief language overview (slide 6 of 10)

Pattern matching

Based on Unix/Linux and Awk style regular expression

`=~` Main pattern matching (binding) operator.

Examples:

```
$answer = "Y";  # Can be 'y', 'Y', 'Yes', 'YES', even 'Yellow'
if ($answer =~ /^y/i) { print "Yes\n" }
```

Metacharacters:

```
if ($file =~ /\s+/) { print "File contains spaces\n" }
```

Substitution:

```
$switch =~ s/on/off/; # Substitute on to off.
```

Advanced:

```
$ora_error =~ /ORA-(01345|01110).+?['"](.+?)['"]/gms
$ora_data_file{$1} = $2;
```

Brief language overview (slide 7 of 10)

Quotation marks – single and double quotes

" (double quotes) do *variable interpolation (variable substitution)*

' (single quotes) suppress *variable interpolation (no variable substitution)*

```
$date = "11 October 2011";
$today = "Double quotes: Today is $date\n";
print $today;
Double quotes: Today is 11 October 2011
$today = 'Single quotes: Today is $date\n';
print $today;
Single quotes: Today is $date\n
```

```
Escape character \
$today = "\'Today is $date\'\n";
print $today;
'Today is 11 October 2011'
```

Brief language overview (slide 8 of 10)

Reading files

```
my $file = 'c:\temp\logfile.txt';
open(FILE, "<", $file) or die "Cannot open $file.\n$!\n";
while (<FILE>) {
    chomp; # Gets rid of white spaces and line breaks
     # do stuff on each line. $_ contains each line
    print "$_\n";
}
close (FILE);
```

Writing files

```
open(FILE, ">", $file)  # Create new file and write to it
open(FILE, ">>", $file) # Append to existing file or create it
```

FILE is filehandle and can be any name. Standard is to use uppercase names.

Perl Question 2

Brief language overview (slide 9 of 10)

Dbvisit DATABASE TECHNOLOGY

Functions / subroutines

```
sub set_oracle_home {
    my $db          = shift;
    my $oracle_home = shift;
    #####
    # Function logic starts here
    #####
    print "db = $db\n";
    print "oracle_home = $oracle_home\n";
}
```

Calling the function:

```
set_oracle_home ("PROD1", '/oracle/product/11g/');
```

What if we want to call:

```
set_oracle_home ('/oracle/product/11g/');
```

Then use parameter calling

Brief language overview (slide 10 of 10)

Functions / subroutines (parameter calling)

```
# Declare the function with parameter calling
sub set_oracle_home {
    my %params      = @_;
    my $db          = $params{db};
    my $oracle_home = $params{oracle_home};
    #####
    # Function logic starts here
    #####
    print "db = $db\n";
    print "oracle_home = $oracle_home\n";
}

# Call the function with parameter calling.
set_oracle_home ( db => "PROD1", oracle_home => '/oracle/product/11g/');

# Reverse the parameters still gives the same result.
set_oracle_home (oracle_home => '/oracle/product/11g/', db => "PROD1");
```

Completed – Perl brief language overview in 10 slides!

Final question at the end

Gotchas in Perl

1) `if (!$sequence) { print "sequence is empty($sequence)\n" }`

This applies when $sequence is empty AND $sequence == 0.

Better:

```
if (!defined($sequence)) { print "sequence is empty($sequence)\n" }
```

Or

```
if ($sequence eq "") { print "sequence is empty($sequence)\n" }
```

2) `@sequence = (2,3,4,5);`

i) `$first_one  = @sequence;`

ii) `($first_one) = @sequence;`

```
print "First one: $first_one\n";
```

i) "First one: 4" ← Prints out the number of elements in the array!

ii) "First one: 2" ← Display the first element in the array.

Making Perl portable

File path components:

/ on Linux and Unix

\ on Windows

: Mac

Use `File::Spec` CPAN module to address this and use `catfile` function:

Eg on Linux:

Example: `curdir = /home/users`

```
$new_file = catfile( curdir(), "temp", "dbvisit.trc");
```

Result: `$new_file = /home/users/temp/dbvisit.trc`

Eg on Windows:

Example: `curdir = C:\Documents and Settings\All Users`

```
$new_file = catfile( curdir(), "temp", "dbvisit.trc");
```

Result: `$new_file = C:\Documents and Settings\All Users\temp\dbvisit.trc`

Perl and Oracle: - shell

How it is done in shell script using IO redirect (<<) and *inline-data*:

```
sqlplus -s <<- EOF > /usr/tmp/sqlplus_tmp.log
\/ as sysdba
clear columns
set linesize 10000
set pause off
set verify off
set trimspool on
set pages 0
set feedback off
select member from v\$logfile;
exit
EOF
echo "Output =====>"
cat /usr/tmp/sqlplus_tmp.log
```

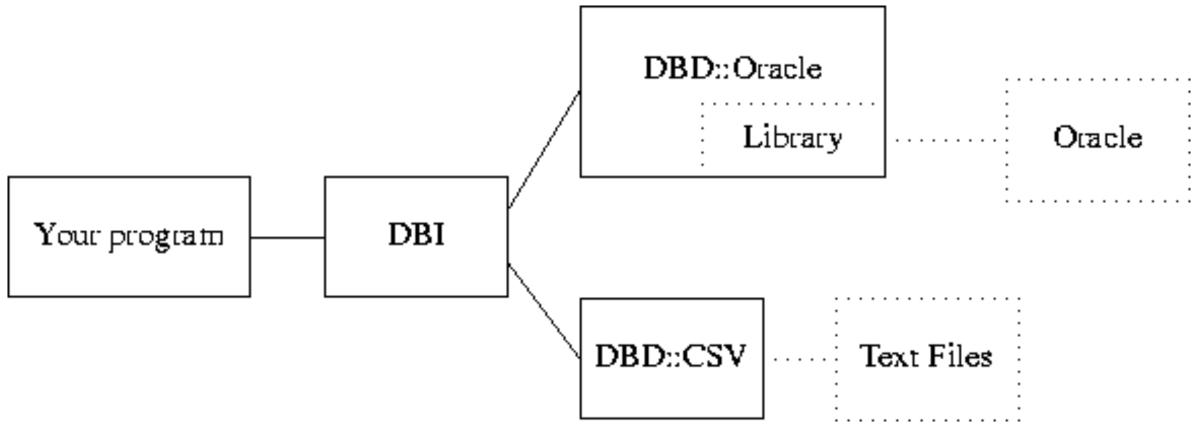
```
Output =====>
/oracle/oradata/dbvisitp/redo03.log
/oracle/oradata/dbvisitp/redo01.log
/oracle/oradata/dbvisitp/redo02.log
```

Perl and Oracle: - Perl

```
#####
# Declaration
#####
my $connect = qq("\/ as sysdba");
my $sql_extra1 = "clear columns
set linesize 10000
set pause off
set verify off
set trimspool on
set pages 0
set feedback off\n";
#####
# Create the SQL command file
#####
open (SQL,">","/usr/tmp/sqlplus_tmp.sql") or die "cannot open file\n";
print SQL $sql_extra1;
print SQL "select member from v\$logfile;\n";
print SQL "exit";
close (SQL);
#####
# Run the command and capture the output. qx() is equivalent to backticks:
``
#####
@sqlplus_output = qx(sqlplus -s $connect \@/usr/tmp/sqlplus_tmp.sql);
print "Output =====>\n";
foreach (@sqlplus_output){
    chomp;
    print "$_\n";
}
```

Perl and Oracle: - DBD::Oracle

Using DBD::Oracle - Oracle database driver for the DBI module.



```
use DBI;
$dbh = DBI->connect("dbi:Oracle:$dbname", $user, $passwd);
my $SEL = "select member from v\$logfile;";
my $sth = $db->prepare($SEL);
$sth->execute();
@data = $sth->fetchrow_array());
```

The advantage with this method is that you have the result set already in a Perl variable. With previous sqlplus method you always have to parse the resulting set as the output is unformatted text.

Popular CPAN modules:

MIME::Lite	Send email on all platforms.
Log::Log4perl	Flexible logging for debugging and log files.
Number::Format	Format numbers.
File::Basename	Parse file paths into directory, filename and suffix.
File::Temp	Automatically find the temp system dir.
Sys::Hostname	Try every conceivable way to get hostname.
File::Spec	Portably perform operations on file names.
Template::Toolkit	HTML template Processing System.
DBD::Oracle	Oracle database driver for the DBI module.
MooseX::Declare	The postmodern object system for Perl 5.

Best way to learn Perl

Set your self a small project

- Check alert log for errors and email them
- Check rman backups and email if errors

Best Perl book:

Programming Perl

By Larry Wall,

Tom Christiansen and Job Orwant

There's More Than One Way To Do It
3rd Edition
Revised & Updated
Programming
Perl
O'REILLY
Larry Wall, Tom Christiansen & Jon Orwant

Small Perl project to take away

Shows free space in Oracle tablespaces and filesystem.
Platform independant!

```
perl FreeSpace.pl XE C:\oracle\xe\app\oracle\product
\10.2.0\server
==>Database: XE
Tablespace           Used Mb      Free Mb     Total Mb   Pct Free
------------- ----------- ----------- ----------- ----------
SYSTEM                   447           3         450          1
SYSAUX                   430          10         440          2
USERS                      7          93         100         93
UNDO                       5         205         210         98

OS       : MSWin32
Hostname: laptop03
Filesystem C:\
        Total Mb: 139,746.99
        Used Mb : 117,393.2
        Free Mb : 22,353.8
        Pct Free: 84
Filesystem D:\
        Total Mb: 10,240
        Used Mb : 3,890.92
        Free Mb : 6,349.08
        Pct Free: 38
```

Small Perl project to take away

To download code:

- http://www.dbvisit.com/oow2011.php

Add the following functionality (homework):

1. Add total Database size.
2. Supply threshold in % for tablespace and filesystem and send emails if thresholds have exceeded.
3. Only have to supply the Oracle SID for Linux/Unix. Work our the ORACLE_HOME from /etc/oratab or /var/opt/oracle/oratab

Perl links

- www.perl.org
- search.cpan.org
- perl6.org
- perlmonks.org
- strawberryperl.com
- activestate.com
- perldoc.perl.org/perlintro.html

Perl Question 3

Dbvisit

THE SMART ALTERNATIVE

Questions?

Try Perl on your next project

Meet the Dbvisit team

in Exhibition Hall