D

The Perl Standard Modules

The following appendix lists and describes the standard and pragmatic modules that are installed with **Perl 5.6**. For this reference, they have been ordered alphabetically by group. Note that these modules' names are case sensitive and are given as they should be written in a `use` statement. For more detialed information, you should turn to the module documentation installed with your version of Perl.

Pragmatic Modules

Using pragmatic modules affects the compilation of your perl programs. These modules are lexically scoped and thus to `use` or to uninclude them with `no` like so

```
use attrs;
use warnings;
no integer;
no diagnostics;
```

is effective only for the duration of the block in which the declaration was made. Furthermore, these declarations may be reversed within any inner blocks in the program.

Name of Module	Function
`attributes`	Gets or sets the attribute values for a subroutine or variable.
`attrs`	Gets or sets the attribute values for a subroutine.
	Deprecated in Perl 5.6 in favour of `attributes`.
`autouse`	Moves the inclusion of modules into a program from compile time to runtime. Specifically, it postpones the module's loading until one of its functions is called.
`base`	Takes a list of modules, `requires` them and then pushes them onto `@ISA`. Essentially, it will establish an IS-A relationship with these classes at compile time.

Table continued on following page

Name of Module	Function
blib	Used on the command line as -Mblib switch to test your scripts against an uninstalled version of the package named after the switch.
caller	Causes program to inherit the pragmatic attributes of the program which has called it.
charnames	Allows you to specify a long name for a given string literal escape.
constant	Allows you to define constants as a name=>value pair.
diagnostics	Returns verbose output when errors occur at runtime. This verbose output consists of the message that perl would normally give plus any accompanying text that that error contained in the perldiag manpage. See Chapter 9 for more on diagnostics.
fields	Takes a list of valid class fields for the package and enables them at compile time.
filetest	Changes the operation of the -r -w -x -R -W and -X file test operators.
integer	Changes the mathematical operators in a program to work with integers only and not floating point numbers.
less	Currrently not implemented.
lib	Adds the listed directories to @INC.
locale	Enables\disables POSIX locales for built-in operations.
ops	Restricts potentially harmful operations occuring during compile time.
overload	Allows you to overload built-in perl operations with your own subroutines.
re	Allows you to alter the way regular expressions behave.
sigtrap	Enables some simple signal handlers.
strict	Enforces the declaration of variables before their use. See Chapter 9 for more on strict.
subs	Allows you to predeclare subroutine names.
utf8	Enables\ disables Unicode support. Note that at the time of writing, Unicode support in Perl was incomplete.
vars	Allows you to predeclare global variable names.
warnings	Switches on the extra syntactic error warning messages.

Standard Modules

The standard modules are the group of modules that are installed with your distribution of Perl.

A

Name of Module	Function
AnyDBM_File	Acts as a universal virtual base class for those wanting to access any of the fivetypes of DBM file.
AutoLoader	Works with Autosplit to delay the loading of subroutines into the program until they are called. These subroutines are defined following the __END__ token in a package file.
AutoSplit	Splits a program into files suitable for autoloading or selfloading.

B

Name of Module	Function
B	The Perl compiler module.
B::Asmdata	Contains autogenerated data about Perl ops used in the generation of bytecode.
B::Assembler	Assembles Perl bytecode for use elsewhere.
B::Bblock	Used by B::CC to walk through 'basic blocks' of code.
B::Bytecode	Compiler backend for generating Perl bytecode.
B::C	Compiler backend for generating C source code.
B::CC	Compiler backend for generating optimized C source code.
B::Debug	Walks the Perl syntax tree, printing debug info about ops
B::Deparse	Compiler backend for generating Perl source code from compiled
B::Disassembler	Disassembles Perl bytecode back to Perl source
B::Graph	Compiler backend for generating graph-description documents that show the program's structure.
B::Lint	Module to catch dubious constructs
B::Showlex	Shows the file-scope variables for a file or the lexical variables for a subroutine if one is specified.
B::Stackobj	Helper module for CC backend.
B::Stash	Shows what stashes are loaded
B::Terse	Walks the Perl syntax tree, printing terse info about ops
B::Xref	Compiler backend for generating cross-reference reports
Benchmark	Contains a suite of routines that let you benchmark your code
ByteLoader	Used to load byte-compiled Perl code

C

Name of Module	Function
CGI	The base class that provides the basic functionality for generating web content and CGI scripting. See Chapter 13 for more.
CGI::Apache	Backward compatibility module.
	Deprecated in Perl 5.6.
CGI::Carp	Holds the equivalent of the Carp module's error logging functions CGI routines for writing time-stamped entries to the HTTPD (or other) error log.
CGI::Cookie	Allows access and interaction with Netscape cookies.
CGI::Fast	Allows CGI access and interaction to a FastCGI web server.
CGI::Pretty	Generates 'pretty' HTML code on server in place of slightly less pretty HTML written in the CGI script file.
CGI::Push	Provides a CGI interface to server-side push functionality. For example, as used with channels.
CGI::Switch	Backward compatibility module.
	Deprecated in Perl 5.6.
CPAN	Provides you with the functionality to query, download, and build Perl modules from any of the CPAN mirrors.
CPAN::FirstTime	Utility for CPAN::Config to ask a few questions about the system and then write a config file.
CPAN::Nox	As CPAN module, but doesn't use any compiled extensions during its own execution.
Carp	Provides warn() and die() functionality with the added ability to say in which module something failed and what it was.
Carp::Heavy	Carp guts. For internal use only.
Class::Struct	Lets you declare C-style struct-like complex datatypes and manipulate them accordingly.
Config	Allows access to the options and settings used by Configure to build this installation of Perl.
Cwd	Gets the pathname of the current working directory

D

Name of Module	Function
DB	Programmatic interface to the Perl debugger's API (Application Programing Interface). N.B.: This may change.
DB_File	Provides an interface for access to Berkeley DB versions 1.x. Note that you can access versions 2.x and 3.x of Berkeley DB with this module but will have only the version 1.x functionality.

Name of Module	Function
Data::Dumper	Returns a stringified version of the contents of an object, given a reference to it.
Devel::DProf	A perl code profiler. Generates information on the frequency of calls to subroutines and on the speediness of the subroutines themselves.
Devel::Peek	A debugging tool for those trying to write C programs that interconnect with Perl programs.
Devel::SelfStubber	Stub generator for a SelfLoading module.
DirHandle	Provides an alternative set of functions to opendir(), closedir(), readdir() and rewinddir().
Dumpvalue	Dumps info about Perl data to the screen.
DynaLoader	Dynamically loads C libraries when required into your Perl code.

E

Name of Module	Function
English	Allows you to call Perl's special variables (see Appendix B) by their 'English' names.
Env	Allows you to access the key\value pairs in the environment hash %ENV as arrays or scalar values.
Errno	Exports (to your code) the contents of the errno.h include file. This contains all the defined error constants on your system.
Exporter	Implements the default import method for modules.
Exporter::Heavy	The internals of the Exporter module.
ExtUtils::Command	Contains equivalents of the common UNIX system commands for Windows users.
ExtUtils::Embed	Contains utilities for embedding a Perl interpreter into your C/C++ programs.
ExtUtils::Install	Contains three functions for installing, uninstalling and installing-as-autosplit/autoload, programs.
ExtUtils::Installed	Keeps track of what modules are and are not installed.
ExtUtils::Liblist	Determine which libraries should be used in an install and how to use them and sends its finding for inclusion in a Makefile.
ExtUtils::MM_Cygwin	Contains methods to override those in ExtUtils::MM_Unix when ExtUtils::MakeMaker is used on a Cygwin system.
ExtUtils::MM_OS2	Contains methods to override those in ExtUtils::MM_Unix when ExtUtils::MakeMaker is used on a OS\2 system.

Table continued on following page

Name of Module	Function
ExtUtils::MM_Unix	Contains the methods used by ExtUtils::MakeMaker to work.
ExtUtils::MM_VMS	Contains methods to override those in ExtUtils::MM_Unix when ExtUtils::MakeMaker is used on a VMS system.
ExtUtils::MM_Win32	Contains methods to override those in ExtUtils::MM_Unix when ExtUtils::MakeMaker is used on a Windows system.
ExtUtils::MakeMaker	Used to create makefiles for an extension module.
ExtUtils::Manifest	Utilities to write and check a MANIFEST file
ExtUtils::Miniperl	Contains one function to write perlmain.c, a bootstrapper between Perl and C libraries.
ExtUtils::Mkbootstrap	Contains one function to write a bootstrap file for use by DynaLoader
ExtUtils::Mksymlists	Contains one function to write linker options files for dynamic extension
ExtUtils::Packlist	Contains a standard .packlist file manager.
ExtUtils::testlib	Adds blib/* directories to @INC

F

Name of Module	Function
Fatal	Provides a way to replace functions which return false with functions that raise an exception if not successful.
Fcntl	Loads the libc fcntl.h defines.
File::Basename	Provides functions that work on a file's full path name
File::CheckTree	Allows you to specify file tests to be made on directories and files within a directory all at once.
File::Compare	Compares the contents of two files.
File::Copy	Copies files or directories.
File::DosGlob	Implements DOS-like globbing but also accepts wildcards in directory components.
File::Find	Searches \ traverses a file tree for requested file.
File::Glob	Implements the FreeBSD glob routine.
File::Path	Creates or deletes a series of directories.
File::Spec	Group of functions to work on file properties and paths.

Name of Module	Function
File::Spec::Functions	Support module for File::Spec
File::Spec::Mac	Contains methods to override those in File::Spec::Unix when File::Spec is used on a MacOS system.
File::Spec::OS2	Contains methods to override those in File::Spec::Unix when File::Spec is used on a OS/2 system.
File::Spec::Unix	Methods used by File::Spec
File::Spec::VMS	Contains methods to override those in File::Spec::Unix when File::Spec is used on a VMS system.
File::Spec::Win32	Contains methods to override those in File::Spec::Unix when File::Spec is used on a Win32 system.
File::stat	A by-name interface to Perl's built-in stat() functions
FindBin	Locates the directory holding the currently running Perl program.
FileCache	Allows you to keep more files open than the system allows.
FileHandle	Provides an OO-style implementation of filehandles.

G

Name of Module	Function
GDBM_File	Provides an interface for access and make use of the GNU Gdbm library.
Getopt::Long	Enables the parsing of long switch names on the command line. See Chapter 9 for more on this.
Getopt::Std	Enables the parsing of single-character switches and clustered switches on the command line. See Chapter 9 for more on this.

I

Name of Module	Function
I18N::Collate	Compares 8-bit scalar data according to the current locale. **Deprecated in Perl 5.004.**
IO	Front-end to load the IO modules listed below.
IO::Dir	Provides an OO-style implementation for directory handles.
IO::File	Based on FileHandle, it provides an OO-style implementation of filehandles.

Table continued on following page

Name of Module	Function
IO::Handle	Provides an OO-style implementation for I/O handles.
IO::Pipe	Provides an OO-style implementation for pipes.
IO::Poll	Provides an OO-style implementation to system poll calls.
IO::Seekable	Provides seek(), sysseek() and tell() methods for I/O objects.
IO::Select	Provides an OO-style implementation for the select system call
IO::Socket	Provides an OO-style implementation for socket communications
IO::Socket::INET	Provides an OO-style implementation for AF_INET domain sockets
IO::Socket::UNIX	Provides an OO-style implementation for AF_UNIX domain sockets
IPC::Msg	Implements a System V Messaging IPC object class.
IPC::Open2	Opens a process for both reading and writing
IPC::Open3	Opens a process for reading, writing, and error handling
IPC::Semaphore	Implements a System V Semaphore IPC object class.
IPC::SysV	Exports all the constants needed by System V IPC calls as defined in your system's libraries.

M

Name of Module	Function
Math::BigFloat	Enables the storage of arbitrarily long floating-point numbers.
Math::BigInt	Enables the storage of arbitrarily long integers.
Math::Complex	Enables work with complex numbers and associated mathematical functions
Math::Trig	Provides all the trigonometric functions not defined in the core of Perl.

N

Name of Module	Function
NDBM_File	Provides access to 'new' DBM files via tied hashes.
Net::Ping	Provides the ability to ping a remote machine via TCP, UDP and ICMP protocols.
Net::hostent	Replaces the core gethost*() functions with those that return Net::hostent objects.
Net::netent	Replaces the core getnet*() functions with those that return Net::netent objects.

Name of Module	Function
Net::protoent	Replaces the core getproto*() functions with those that return Net::protoent objects.
Net::servent	Replaces the core getserv*() functions with those that return Net::servent objects.

O

Name of Module	Function
O	This is the generic frontend for the Perl compiler. The backends in the B module group are all addressed with this.
Opcode	Allows you to disable named opcodes when compiling Perl code

P

Name of Module	Function
Pod::Checker	Provides a syntax error checker for pod documents. Note that this was still in beta at the time of publication.
Pod::Html	Pod to HTML converter.
Pod::InputObjects	A set of objects that can be used to represent pod files.
Pod::Man	Pod to *roff converter.
Pod::Parser	Base class for creating POD filters and translators.
Pod::Select	Used to extract selected sections of POD from input
Pod::Text	Pod to formatted ASCII text converter.
Pod::Text::Color	Pod to formatted, colored ASCII text converter.
Pod::Usage	Print a usage message from embedded pod documentation
POSIX	Provides access to (nearly) all the functions and identifiers named in the POSIX international standard 1003.1.

S

Name of Module	Function
Safe	Creates a number of 'safe' compartments in memory in which perl code can be tested and the functions for this testing.
SDBM_File	Provides access to sdbm files via tied hashes.
Search::Dict	Provides function to look for a key in a dictionary file.

Table continued on following page

Name of Module	Function
SelectSaver	Selects a filehandle on creation, saves it and restores it on destruction.
SelfLoader	As Autoloader, works with Autosplit to delay the loading of subroutines into the program until they are called. These subroutines are defined following the __DATA__ token in a package file.
Shell	Allows shell commands to be run transparently within perl programs.
Socket	Imports the defines from libc's socket.h header file and makes available some network manipulation fucntions.
Symbol	Qualifies variable names and creates anonymous globs.
Sys::Hostname	Makes several attempts to get the system hostname and then cahces the result.
Sys::Syslog	Perl's interface to the libc syslog(3) calls

T

Name of Module	Function
Term::Cap	Provides the interface to a terminal capability database.
Term::Complete	Provides word completion on the word list in an array.
Term::ReadLine	Provides access to various 'readline' packages.
Test	Provides a simple framework for writing test scripts
Test::Harness	Implements a test harness to run a series of test scripts and return results.
Text::Abbrev	Takes a list and returns a hash containing the elements of the list as the values and unambiguous abbreviations of each element as their respective keys.
Text::ParseWords	Provides functions for parsing a text file into an array of tokens or an array of arrays.
Text::Soundex	Implementation of the Soundex Algorithm.
Text::Tabs	Works through lines of text replacing tabs with spaces, or if space-saving, replacing spaces with tabs if there are none in the text.
Text::Wrap	Simple paragraph formatter. Takes text, wraps lines around text boundaries and controls the indenting of the text.
Tie::Array	Base class for tied arrays.
Tie::Handle	Base class definitions for tied handles.
Tie::Hash	Base class definitions for tied hashes.

Name of Module	Function
`Tie::RefHash`	Allows you to use references as the keys in a hash if it is tied to this module.
`Tie::Scalar`	Base class definitions for tied scalars.
`Tie::SubstrHash`	Allows you to rigidly define key and value lengths within the hash for the entire time it is tied to this module.
`Time::gmtime`	Object-based interface to Perl's built-in `gmtime()` function.
`Time::Local`	Provides efficient conversion functions between GMT and local time.
`Time::localtime`	Object-based interface to Perl's built-in `localtime()` function.
`Time::tm`	Internal object used by `Time::gmtime` and `Time::localtime`

U

Name of Module	Function
`UNIVERSAL`	The base class for ALL classes (blessed references)
`User::grent`	Object-based interface to Perl's built-in `getgr*()` functions
`User::pwent`	Object-based interface to Perl's built-in `getpw*()` functions