NAME

perl593delta - what is new for perl v5.9.3

DESCRIPTION

This document describes differences between the 5.9.2 and the 5.9.3 development releases. See *perl590delta*, *perl591delta* and *perl592delta* for the differences between 5.8.0 and 5.9.2.

Incompatible Changes

Parsing of -f _

The identifier _ is now forced to be a bareword after a filetest operator. This solves a number of misparsing issues when a global _ subroutine is defined.

mkdir()

`mkdir()` without arguments now defaults to `$_`.

Magic goto and eval

The construct `eval { goto &foo }` is now disallowed. (Note that the similar construct, but with `eval("")` instead, was already forbidden.)

$# has been removed

The deprecated $# variable (output format for numbers) has been removed. A new warning, `$# is no longer supported`, has been added.

:unique

The `:unique` attribute has been made a no-op, since its current implementation was fundamentally flawed and not threadsafe.

Scoping of the sort pragma

The `sort` pragma is now lexically scoped. Its effect used to be global.

Core Enhancements

The feature pragma

The `feature` pragma is used to enable new syntax that would break Perl's backwards-compatibility with older releases of the language. It's a lexical pragma, like `strict` or `warnings`.

Currently the following new features are available: `switch` (adds a switch statement), `~~` (adds a Perl 6-like smart match operator), `say` (adds a `say` built-in function), and `err` (adds an `err` keyword). Those features are described below.

Note that `err` low-precedence defined-or operator used to be enabled by default (although as a weak keyword, meaning that any function would override it). It's now only recognized when explicitly turned on (and is then a regular keyword).

Those features, and the `feature` pragma itself, have been contributed by Robin Houston.

Switch and Smart Match operator

Perl 5 now has a switch statement. It's available when `use feature 'switch'` is in effect. This feature introduces three new keywords, `given`, `when`, and `default`:

```
    given ($foo) {
 when (/^abc/) { $abc = 1; }
 when (/^def/) { $def = 1; }
 when (/^xyz/) { $xyz = 1; }
 default { $nothing = 1; }
    }
```

A more complete description of how Perl matches the switch variable against the `when` conditions is given in *"Switch statements" in perlsyn*.

This kind of match is called *smart match*, and it's also possible to use it outside of switch statements, via the new ~~ operator (enabled via the `use feature '~~'` directive). See *"Smart matching in detail" in perlsyn*.

say()

say() is a new built-in, only available when `use feature 'say'` is in effect, that is similar to print(), but that implicitly appends a newline to the printed string. See *"say" in perlfunc*.

CLONE_SKIP()

Perl has now support for the `CLONE_SKIP` special subroutine. Like `CLONE`, `CLONE_SKIP` is called once per package; however, it is called just before cloning starts, and in the context of the parent thread. If it returns a true value, then no objects of that class will be cloned. See *perlmod* for details. (Contributed by Dave Mitchell.)

${^CHILD_ERROR_NATIVE}

A new internal variable, `${^CHILD_ERROR_NATIVE}`, gives the native status returned by the last pipe close, backtick command, successful call to wait() or waitpid(), or from the system() operator. See *perlrun* for details. (Contributed by Gisle Aas.)

Assertions

The support for assertions, introduced in perl 5.9.0, has been improved. The syntax for the `-A` command-line switch has changed; it now accepts an optional module name, defaulting to `assertions::activate`. See *assertions* and *perlrun*. (Contributed by Salvador Fandiño García.)

Unicode Character Database 4.1.0

The copy of the Unicode Character Database included in Perl 5.9 has been updated to 4.1.0.

no VERSION

You can now use `no` followed by a version number to specify that you want to use a version of perl older than the specified one.

Recursive sort subs

You can now use recursive subroutines with sort(), thanks to Robin Houston.

Effect of pragmas in eval

The compile-time value of the `%^H` hint variable can now propagate into eval("")uated code. This makes it more useful to implement lexical pragmas.

As a side-effect of this, the overloaded-ness of constants now propagates into eval("").

New -E command-line switch

-E is equivalent to **-e**, but it implicitly enables all optional features (like `use feature ":5.10"`).

chdir, chmod and chown on filehandles

`chdir`, `chmod` and `chown` can now work on filehandles as well as filenames, if the system supports respectively `fchdir`, `fchmod` and `fchown`, thanks to a patch provided by Gisle Aas.

OS groups

`$(` and `$)` now return groups in the order where the OS returns them, thanks to Gisle Aas. This wasn't previously the case.

Modules and Pragmata

New Core Modules

- A new pragma, `feature`, has been added; see above in *Core Enhancements*.
- `assertions::compat`, also available on CPAN, allows the use of assertions on perl versions prior to 5.9.0 (that is the first one to natively support them).
- `Math::BigInt::FastCalc` is an XS-enabled, and thus faster, version of `Math::BigInt::Calc`.
- `Compress::Zlib` is an interface to the zlib compression library. It comes with a bundled version of zlib, so having a working zlib is not a prerequisite to install it. It's used by `Archive::Tar` (see below).
- `IO::Zlib` is an `IO::`-style interface to `Compress::Zlib`.
- `Archive::Tar` is a module to manipulate `tar` archives.
- `Digest::SHA` is a module used to calculate many types of SHA digests, has been included for SHA support in the CPAN module.
- `ExtUtils::CBuilder` and `ExtUtils::ParseXS` have been added.

Utility Changes

ptar

`ptar` is a pure perl implementation of `tar`, that comes with `Archive::Tar`.

ptardiff

`ptardiff` is a small script used to generate a diff between the contents of a tar archive and a directory tree. Like `ptar`, it comes with `Archive::Tar`.

shasum

This command-line utility, used to print or to check SHA digests, comes with the new `Digest::SHA` module.

h2xs enhancements

`h2xs` implements a new option `--use-xsloader` to force use of `XSLoader` even in backwards compatible modules.

The handling of authors' names that had apostrophes has been fixed.

Any enums with negative values are now skipped.

perlivp enhancements

`perlivp` no longer checks for *.*ph* files by default. Use the new `-a` option to run *all* tests.

Documentation

Perl Glossary

The *perlglossary* manpage is a glossary of terms used in the Perl documentation, technical and otherwise, kindly provided by O'Reilly Media, Inc.

perltodo now lists a rough roadmap to Perl 5.10.

Performance Enhancements

XS-assisted SWASHGET

Some pure-perl code that perl was using to retrieve Unicode properties and transliteration mappings has been reimplemented in XS.

Constant subroutines

The interpreter internals now support a far more memory efficient form of inlineable constants. Storing a reference to a constant value in a symbol table is equivalent to a full typeglob referencing a constant subroutine, but using about 400 bytes less memory. This proxy constant subroutine is automatically upgraded to a real typeglob with subroutine if necessary. The approach taken is analogous to the existing space optimisation for subroutine stub declarations, which are stored as plain scalars in place of the full typeglob.

Several of the core modules have been converted to use this feature for their system dependent constants - as a result `use POSIX;` now takes about 200K less memory.

PERL_DONT_CREATE_GVSV

The new compilation flag `PERL_DONT_CREATE_GVSV`, introduced as an option in perl 5.8.8, is turned on by default in perl 5.9.3. It prevents perl from creating an empty scalar with every new typeglob. See *perl588delta* for details.

Weak references are cheaper

Weak reference creation is now *O(1)* rather than *O(n)*, courtesy of Nicholas Clark. Weak reference deletion remains *O(n)*, but if deletion only happens at program exit, it may be skipped completely.

sort() enhancements

Salvador Fandiño provided improvements to reduce the memory usage of `sort` and to speed up some cases.

Installation and Configuration Improvements

Compilation improvements

Parallel makes should work properly now, although there may still be problems if `make test` is instructed to run in parallel.

Building with Borland's compilers on Win32 should work more smoothly. In particular Steve Hay has worked to side step many warnings emitted by their compilers and at least one C compiler internal error.

Perl extensions on Windows now can be statically built into the Perl DLL, thanks to a work by Vadim Konovalov.

New Or Improved Platforms

Perl is being ported to Symbian OS. See *perlsymbian* for more information.

The VMS port has been improved. See *perlvms*.

DynaLoader::dl_unload_file() now works on Windows.

Portability of Perl on various recent compilers on Windows has been improved (Borland C++, Visual C++ 7.0).

New probes

`Configure` will now detect `clearenv` and `unsetenv`, thanks to a patch from Alan Burlison. It will also probe for `futimes` (and use it internally if available), and whether `sprintf` correctly returns the length of the formatted string.

Module auxiliary files

README files and changelogs for CPAN modules bundled with perl are no longer installed.

Selected Bug Fixes

defined $$x

`use strict "refs"` was ignoring taking a hard reference in an argument to defined(), as in :

```
use strict "refs";
my $x = "foo";
if (defined $$x) {...}
```

This now correctly produces the run-time error `Can't use string as a SCALAR ref while "strict refs" in use`. (However, `defined @$foo` and `defined %$foo` are still allowed. Those constructs are discouraged anyway.)

Calling CORE::require()

CORE::require() and CORE::do() were always parsed as require() and do() when they were overridden. This is now fixed.

Subscripts of slices

You can now use a non-arrowed form for chained subscripts after a list slice, like in:

```
({foo => "bar"})[0]{foo}
```

This used to be a syntax error; a `->` was required.

Remove over-optimisation

Perl 5.9.2 introduced a change so that assignments of `undef` to a scalar, or of an empty list to an array or a hash, were optimised out. As this could cause problems when `goto` jumps were involved, this change was backed out.

sprintf() fixes

Using the sprintf() function with some formats could lead to a buffer overflow in some specific cases. This has been fixed, along with several other bugs, notably in bounds checking.

In related fixes, it was possible for badly written code that did not follow the documentation of `Sys::Syslog` to have formatting vulnerabilities. `Sys::Syslog` has been changed to protect people from poor quality third party code.

no warnings 'category' works correctly with -w

Previously when running with warnings enabled globally via `-w`, selective disabling of specific warning categories would actually turn off all warnings. This is now fixed; now `no warnings 'io';` will only turn off warnings in the `io` class. Previously it would erroneously turn off all warnings.

Smaller fixes

- `FindBin` now works better with directories where access rights are more restrictive than usual.
- Several memory leaks in ithreads were closed. Also, ithreads were made less memory-intensive.
- Trailing spaces are now trimmed from `$!` and `$^E`.
- Operations that require perl to read a process' list of groups, such as reads of `$(` and `$)`, now dynamically allocate memory rather than using a fixed sized array. The fixed size array could cause C stack exhaustion on systems configured to use large numbers of groups.
- `PerlIO::scalar` now works better with non-default `$/` settings.
- The `x` repetition operator is now able to operate on `qw//` lists. This used to raise a syntax error.

- The debugger now traces correctly execution in eval(""")uated code that contains #line directives.
- The value of the open pragma is no longer ignored for three-argument opens.
- Perl will now use the C library calls unsetenv and clearenv if present to delete keys from %ENV and delete %ENV entirely, thanks to a patch from Alan Burlison.

More Unicode Fixes

- chr() on a negative value now gives \x{FFFD}, the Unicode replacement character, unless when the bytes pragma is in effect, where the low eight bytes of the value are used.
- Some case insensitive matches between UTF-8 encoded data and 8 bit regexps, and vice versa, could give malformed character warnings. These have been fixed by Dave Mitchell and Yves Orton.
- lcfirst and ucfirst could corrupt the string for certain cases where the length UTF-8 encoding of the string in lower case, upper case or title case differed. This was fixed by Nicholas Clark.

New or Changed Diagnostics

Attempt to set length of freed array

This is a new warning, produced in situations like the following one:

```
$r = do {my @a; \$#a};
$$r = 503;
```

Non-string passed as bitmask

This is a new warning, produced when number has been passed as a argument to select(), instead of a bitmask.

```
# Wrong, will now warn
$rin = fileno(STDIN);
($nfound,$timeleft) = select($rout=$rin, undef, undef, $timeout);

# Should be
$rin = '';
vec($rin,fileno(STDIN),1) = 1;
($nfound,$timeleft) = select($rout=$rin, undef, undef, $timeout);
```

Search pattern not terminated or ternary operator parsed as search pattern

This syntax error indicates that the lexer couldn't find the final delimiter of a ?PATTERN? construct. Mentioning the ternary operator in this error message makes syntax diagnostic easier.

"%s" variable %s masks earlier declaration

This warning is now emitted in more consistent cases; in short, when one of the declarations involved is a my variable:

```
my $x;   my $x; # warns
my $x;  our $x; # warns
our $x;  my $x; # warns
```

On the other hand, the following:

```
our $x; our $x;
```

now gives a `"our" variable %s redeclared` warning.

readdir()/closedir()/etc. attempted on invalid dirhandle

These new warnings are now emitted when a dirhandle is used but is either closed or not really a dirhandle.

Changed Internals

In general, the source code of perl has been refactored, tied up, and optimized in many places. Also, memory management and allocation has been improved in a couple of points.

Andy Lester supplied many improvements to determine which function parameters and local variables could actually be declared `const` to the C compiler. Steve Peters provided new `*_set` macros and reworked the core to use these rather than assigning to macros in LVALUE context.

Dave Mitchell improved the lexer debugging output under `-DT`.

A new file, *mathoms.c*, has been added. It contains functions that are no longer used in the perl core, but that remain available for binary or source compatibility reasons. However, those functions will not be compiled in if you add `-DNO_MATHOMS` in the compiler flags.

The `AvFLAGS` macro has been removed.

The `av_*()` functions, used to manipulate arrays, no longer accept null `AV*` parameters.

B:: modules inheritance changed

The inheritance hierarchy of `B::` modules has changed; `B::NV` now inherits from `B::SV` (it used to inherit from `B::IV`).

Reporting Bugs

If you find what you think is a bug, you might check the articles recently posted to the comp.lang.perl.misc newsgroup and the perl bug database at http://bugs.perl.org/ . There may also be information at http://www.perl.org/ , the Perl Home Page.

If you believe you have an unreported bug, please run the **perlbug** program included with your release. Be sure to trim your bug down to a tiny but sufficient test case. Your bug report, along with the output of `perl -V`, will be sent off to perlbug@perl.org to be analysed by the Perl porting team.

SEE ALSO

The *Changes* file for exhaustive details on what changed.

The *INSTALL* file for how to build Perl.

The *README* file for general stuff.

The *Artistic* and *Copying* files for copyright information.