perl5136delta (source, CPAN)

You are viewing the version of this documentation from Perl 5.14.0. View the latest version

CONTENTS

NAME

perl5136delta - what is new for perl v5.13.6

DESCRIPTION

This document describes differences between the 5.13.5 release and the 5.13.6 release.

If you are upgrading from an earlier release such as 5.13.4, first read perl5135delta, which describes differences between 5.13.4 and 5.13.5.

Core Enhancements

`(?^...)` regex construct added to signify default modifiers

A caret (also called a "circumflex accent") `"^"` immediately following a `"(?"` in a regular expression now means that the subexpression is to not inherit the surrounding modifiers such as `/i`, but to revert to the Perl defaults. Any modifiers following the caret override the defaults.

The stringification of regular expressions now uses this notation. E.g., before, `qr/hlagh/i` would be stringified as `(?i-xsm:hlagh)`, but now it's stringified as `(?^i:hlagh)`.

The main purpose of this is to allow tests that rely on the stringification to not have to change when new modifiers are added. See "Extended Patterns" in perlre.

`"d"`, `"l"`, and `"u"` regex modifiers added

These modifiers are currently only available within a `(?...)` construct.

The `"l"` modifier says to compile the regular expression as if it were in the scope of `use locale`, even if it is not.

The `"u"` modifier says to compile the regular expression as if it were in the scope of a `use feature "unicode_strings"` pragma.

The `"d"` modifier is used to override any `use locale` and `use feature "unicode_strings"` pragmas that are in effect at the time of compiling the regular expression.

See just below and "(?dlupimsx-imsx)" in perlre.

`use feature "unicode_strings"` now applies to some regex matching

Another chunk of the "The "Unicode Bug"" in perlunicode is fixed in this release. Now, regular expressions compiled within the scope of the "unicode_strings" feature will match the same whether or not the target string is encoded in utf8, with regard to `\s`, `\w`, `\b`, and their complements. Work is underway to add the `[[:posix:]]` character classes and case sensitive matching to the control of this feature, but was not complete in time for this dot release.

`\N{...}` now handles Unicode named character sequences

Unicode has a number of named character sequences, in which particular sequences of code points are given names. `\N{...}` now recognizes these. See charnames.

New function `charnames::string_vianame()`

This function is a run-time version of `\N{...}`, returning the string of characters whose Unicode name is its parameter. It can handle Unicode named character sequences, whereas the pre-existing `charnames::vianame()` cannot, as the latter returns a single code point. See charnames.

Reentrant regular expression engine

It is now safe to use regular expressions within `(?{...})` and `(??{...})` code blocks inside regular expressions.

These block are still experimental, however, and still have problems with lexical (`my`) variables, lexical pragmata and abnormal exiting.

Custom per-subroutine check hooks

XS code in an extension module can now annotate a subroutine (whether implemented in XS or in Perl) so that nominated XS code will be called at compile time (specifically as part of op checking) to change the op tree of that subroutine. The compile-time check function (supplied by the extension module) can implement argument processing that can't be expressed as a prototype, generate customised compile-time warnings, perform constant folding for a pure function, inline a subroutine consisting of sufficiently simple ops, replace the whole call with a custom op, and so on. This was previously all possible by hooking the `entersub` op checker, but the new mechanism makes it easy to tie the hook to a specific subroutine. See "cv_set_call_checker" in perlapi.

To help in writing custom check hooks, several subtasks within standard `entersub` op checking have been separated out and exposed in the API.

Return value of `delete $+{...}`

Custom regular expression engines can now determine the return value of `delete` on an entry of `%+` or `%-`.

`keys`, `values` work on arrays

You can now use the `keys`, `values`, `each` builtin functions on arrays (previously you could only use them on hashes). See perlfunc for details. This is actually a change introduced in perl 5.12.0, but it was missed from that release's perldelta.

Incompatible Changes

Stringification of regexes has changed

Default regular expression modifiers are now notated by using `(?^...)`. Code relying on the old stringification will fail. The purpose of this is so that when new modifiers are added, such code will not have to change (after this one time), as the stringification will automatically incorporate the new modifiers.

Code that needs to work properly with both old- and new-style regexes can avoid the whole issue by using (for Perls since 5.9.5):

```
use re qw(regexp_pattern);
my ($pat, $mods) = regexp_pattern($re_ref);
```

where `$re_ref` is a reference to a compiled regular expression. Upon return, `$mods` will be a string containing all the non-default modifiers used when the regular expression was compiled, and `$pattern` the actual pattern.

If the actual stringification is important, or older Perls need to be supported, you can use something like the following:

```
# Accept both old and new-style stringification
my $modifiers = (qr/foobar/ =~ /\Q(?^/) ? '^' : '-xism';
```

And then use `$modifiers` instead of `-xism`.

Regular expressions retain their localeness when interpolated

Regular expressions compiled under `"use locale"` now retain this when interpolated into a new regular expression compiled outside a `"use locale"`, and vice-versa.

Previously, a regular expression interpolated into another one inherited the localeness of the surrounding one, losing whatever state it originally had. This is considered a bug fix, but may trip up code that has come to rely on the incorrect behavior.

Directory handles not copied to threads

On systems that do not have a `fchdir` function, newly-created threads no longer inherit directory handles from their parent threads. Such programs would probably have crashed anyway [perl #75154].

Negation treats strings differently from before

The unary negation operator `-` now treats strings that look like numbers as numbers [perl #57706].

Negative zero

Negative zero (-0.0), when converted to a string, now becomes "0" on all platforms. It used to become "-0" on some, but "0" on others.

If you still need to determine whether a zero is negative, use `sprintf("%g", $zero) =~ /^-/` or the Data::Float module on CPAN.

Performance Enhancements

- The bulk of the `Tie::Hash::NamedCapture` module used to be in the perl core. It has now been moved to an XS module, to reduce the overhead for programs that do not use `%+` or `%-`.
- Eliminate `PL_*` accessor functions under ithreads.
  When `MULTIPLICITY` was first developed, and interpreter state moved into an interpreter struct, thread and interpreter local `PL_*` variables were defined as macros that called accessor functions, returning the address of the value, outside of the perl core. The intent was to allow members within the interpreter struct to change size without breaking binary compatibility, so that bug fixes could be merged to a maintenance branch that necessitated such a size change. However, some non-core code defines `PERL_CORE`, sometimes intentionally to bypass this mechanism for speed reasons, sometimes for other reasons but with the inadvertent side effect of bypassing this mechanism. As some of this code is widespread in production use, the result is that the core **can't** change the size of members of the interpreter struct, as it will break such modules compiled against a previous release on that maintenance branch. The upshot is that this mechanism is redundant, and well-behaved code is penalised by it. Hence it can and should be removed.

Modules and Pragmata

Updated Modules and Pragmata

- `Archive::Extract` has been upgraded from version 0.42 to 0.44
- `Carp` has been upgraded from version 1.18 to 1.19.
  It no longer autovivifies the `*CORE::GLOBAL::caller` glob, something it started doing in 1.18, which was released with perl 5.13.4 [perl #78082]
- `Compress::Raw::Bzip2` has been upgraded from version 2.030 to 2.031
  Updated to use bzip2 1.0.6
- `CPAN` has been upgraded from version 1.94_57 to 1.94_61
- `Data::Dumper` has been upgraded from version 2.128 to 2.129.
  `Dumpxs` no longer crashes with globs returned by `*$io_ref` [perl #72332].
- `Digest::MD5` has been upgraded from version 2.40 to 2.51.
  It is now safe to use this module in combination with threads.
- `File::DosGlob` has been upgraded from version 1.02 to 1.03.
  It allows patterns containing literal parentheses (they no longer need to be escaped). On Windows, it no longer adds an extra *./* to the file names returned when the pattern is a relative glob with a drive specification, like *c:*.pl* [perl #71712].
- `File::Find` has been upgraded from version 1.17 to 1.18.
  It improves handling of backslashes on Windows, so that paths such as *c:\dir\/file* are no longer generated [perl #71710].
- `if` has been upgraded from version 0.05 to 0.06
- `IPC::Cmd` has been upgraded from version 0.60 to 0.64

- `IPC::Open3` has been upgraded from version 1.06 to 1.07.
  The internal `xclose` routine now knows how to handle file descriptors, as documented, so duplicating STDIN in a child process using its file descriptor now works [perl #76474].
- `Locale::Codes` has been upgraded from version 3.13 to 3.14.
- `Locale::Maketext` has been upgraded from version 1.15 to 1.16.
  It fixes an infinite loop in `Locale::Maketext::Guts::_compile()` when working with tainted values (CPAN RT #40727).
  `->maketext` calls will now backup and restore `$@` so that error messages are not suppressed (CPAN RT #34182).
- `Math::BigInt` has been upgraded from version 1.95 to 1.97.
  This prevents `sqrt($int)` from crashing under `use bigrat;` [perl #73534].
- `NEXT` has been upgraded from version 0.64 to 0.65.
- `overload` has been upgraded from version 1.10 to 1.11.
  `overload::Method` can now handle subroutines that are themselves blessed into overloaded classes [perl #71998].
- `PathTools` has been upgraded from version 3.31_01 to 3.34.
- `podlators` has been upgraded from version 2.3.1 to 2.4.0
- `sigtrap` has been upgraded from version 1.04 to 1.05.
  It no longer tries to modify read-only arguments when generating a backtrace [perl #72340].
- `threads` has been upgraded from version 1.77_03 to 1.81_01.
- `threads::shared` has been upgrade from version 1.33_03 to 1.34
- `Unicode::Collate` has been upgraded from version 0.59 to 0.63
  U::C::Locale newly supports locales: ar, be, bg, de__phonebook, hu, hy, kk, mk, nso, om, tn, vi, hr, ig, ru, sq, se, sr, to and uk
- `Unicode::Normalize` has been upgraded from version 1.06 to 1.07
- `B::Deparse` has been upgraded from version 0.98 to 0.99
  B::Deparse now properly handles the code that applies a conditional pattern match against implicit `$_` as it was fixed in [perl #20444].
- `GDBM_File` has been upgraded from version 1.10 to 1.11

Documentation

Changes to Existing Documentation

perlapi

- The documentation for the `SvTRUE` macro was simply wrong in stating that get-magic is not processed. It has been corrected.

Diagnostics

The following additions or changes have been made to diagnostic output, including warnings and fatal error messages. For the complete list of diagnostic messages, see perldiag.

Changes to Existing Diagnostics

- The 'Layer does not match this perl' error message has been replaced with these more helpful messages:
  - PerlIO layer function table size (%d) does not match size expected by this perl (%d)
  - PerlIO layer instance size (%d) does not match size expected by this perl (%d)

  [perl #73754]

Testing

- The script *t/op/threads-dirh.t* has been added, which tests interaction of threads and directory handles.

Platform Support

Platform-Specific Notes

IRIX

Conversion of strings to floating-point numbers is now more accurate on IRIX systems [perl #32380].

Mac OS X

Early versions of Mac OS X (Darwin) had buggy implementations of the `setregid`, `setreuid`, `setrgid` and `setruid` functions, so perl would pretend they did not exist.

These functions are now recognised on Mac OS 10.5 (Leopard; Darwin 9) and higher, as they have been fixed [perl #72990].

OpenVOS

perl now builds again with OpenVOS (formerly known as Stratus VOS) [perl #78132].

VMS

The shortening of symbols longer than 31 characters in the C sources is now done by the compiler rather than by xsubpp (which could only do so for generated symbols in XS code).

Windows

`$Config{gccversion}` is now set correctly when perl is built using the mingw64 compiler from http://mingw64.org [perl #73754].

The build process proceeds more smoothly with mingw and dmake when *C:\MSYS\bin* is in the PATH, due to a `Cwd` fix.

Internal Changes

- See "Regular expressions retain their localeness when interpolated", above.

- The `sv_cmp_flags`, `sv_cmp_locale_flags`, `sv_eq_flags` and `sv_collxfrm_flags` functions have been added. These are like their non-_flags counterparts, but allow one to specify whether get-magic is processed.
  The `sv_cmp`, `sv_cmp_locale`, `sv_eq` and `sv_collxfrm` functions have been replaced with wrappers around the new functions.
- A new `sv_2bool_flags` function has been added.
  This is like `sv_2bool`, but it lets the calling code decide whether get-magic is handled.
  `sv_2bool` is now a macro that calls the new function.
- A new macro, `SvTRUE_nomg`, has been added.
  This is like `SvTRUE`, except that it does not process magic. It uses the new `sv_2bool_flags` function.
- `sv_catsv_flags` no longer calls `mg_get` on its second argument (the source string) if the flags passed to it do not include SV_GMAGIC. So it now matches the documentation.
- A new interface has been added for custom check hooks on subroutines. See "Custom per-subroutine check hooks", above.
- List op building functions have been added to the API. See op_append_elem, op_append_list, and op_prepend_elem.
- The LINKLIST macro, part of op building that constructs the execution-order op chain, has been added to the API.
- Many functions ending with pvn now have equivalent pv/pvs/sv versions.
- The `save_freeop`, `save_op`, `save_pushi32ptr` and `save_pushptrptr` functions have been added to the API.
- The new API function `parse_stmtseq()` parses a sequence of statements, up to closing brace or EOF.

Selected Bug Fixes

- A regular expression match in the right-hand side of a global substitution (`s///g`) that is in the same scope will no longer cause match variables to have the wrong values on subsequent iterations. This can happen when an array or hash subscript is interpolated in the right-hand side, as in `s|(.)|@a{ print($1), /./ }|g` [perl #19078].
- Constant-folding used to cause

  ```
  $text =~ ( 1 ? /phoo/ : /bear/)
  ```

  to turn into

  ```
  $text =~ /phoo/
  ```

  at compile time. Now it correctly matches against `$_` [perl #20444].
- Parsing Perl code (either with string `eval` or by loading modules) from within a `UNITCHECK` block no longer causes the interpreter to crash [perl #70614].
- When `-d` is used on the shebang (`#!`) line, the debugger now has access to the lines of the main program. In the past, this sometimes worked and sometimes did not, depending on what

order things happened to be arranged in memory [perl #71806].

- The `y///` or `tr///` operator now calls get-magic (e.g., the `FETCH` method of a tie) on its left-hand side just once, not twice [perl #76814].
- String comparison (`eq`, `ne`, `lt`, `gt`, `le`, `ge` and `cmp`) and logical not (`not` and `!`) operators no longer call magic (e.g., tie methods) twice on their operands [perl #76814].
  This bug was introduced in an earlier 5.13 release, and does not affect perl 5.12.
- When a tied (or other magic) variable is used as, or in, a regular expression, it no longer has its `FETCH` method called twice [perl #76814].
  This bug was introduced in an earlier 5.13 release, and does not affect perl 5.12.
- The `-C` command line option can now be followed by other options [perl #72434].
- Assigning a glob to a PVLV used to convert it to a plain string. Now it works correctly, and a PVLV can hold a glob. This would happen when a nonexistent hash or array element was passed to a subroutine:

  ```
  sub { $_[0] = *foo }->($hash{key});
  # $_[0] would have been the string "*main::foo"
  ```

  It also happened when a glob was assigned to, or returned from, an element of a tied array or hash [perl #36051].
- Creating a new thread when directory handles were open used to cause a crash, because the handles were not cloned, but simply passed to the new thread, resulting in a double free.
  Now directory handles are cloned properly, on systems that have a `fchdir` function. On other systems, new threads simply do not inherit directory handles from their parent threads [perl #75154].
- The regular expression parser no longer hangs when parsing `\18` and `\88`.
  This bug was introduced in version 5.13.5 and did not affect earlier versions [perl #78058].
- Subroutine redefinition works once more in the debugger [perl #48332].
- The `&` `|` `^` bitwise operators no longer coerce read-only arguments [perl #20661].
- Stringifying a scalar containing -0.0 no longer has the affect of turning false into true [perl #45133].
- Aliasing packages by assigning to globs or deleting packages by deleting their containing stash elements used to have erratic effects on method resolution, because the internal 'isa' caches were not reset. This has been fixed.
- `sort` with a custom sort routine could crash if too many nested subroutine calls occurred from within the sort routine [perl #77930].
  This bug was introduced in an earlier 5.13 release, and did not affect perl 5.12.
- The `eval_sv` and `eval_pv` C functions now set `$@` correctly when there is a syntax error and no `G_KEEPERR` flag, and never set it if the `G_KEEPERR` flag is present [perl #3719].
- Nested `map` and `grep` blocks no longer leak memory when processing large lists [perl #48004].
- Malformed `version` objects no longer cause crashes [perl #78286].
- The interpreter no longer crashes when freeing deeply-nested arrays of arrays. Hashes have not been fixed yet [perl #44225].
- The mechanism for freeing objects in globs used to leave dangling pointers to freed SVs, meaning Perl users could see corrupted state during destruction.

Perl now only frees the affected slots of the GV, rather than freeing the GV itself. This makes sure that there are no dangling refs or corrupted state during destruction.

- The typeglob `*,` , which holds the scalar variable `$,` (output field separator), had the wrong reference count in child threads.
- `splice` now calls set-magic. This means that, for instance, changes made by `splice @ISA` are respected by method calls [perl #78400].
- `use v5.8` no longer leaks memory [perl #78436].
- The XS multicall API no longer causes subroutines to lose reference counts if called via the multicall interface from within those very subroutines. This affects modules like List::Util. Calling one of its functions with an active subroutine as the first argument could cause a crash [perl #78070].

Errata

- Fixed a typo in perl5135delta regarding array slices and smart matching

Acknowledgements

Perl 5.13.6 represents approximately one month of development since Perl 5.13.5 and contains 67920 lines of changes across 566 files from 47 authors and committers:

A. Sinan Unur, Aaron Crane, Alex Davies, Ali Polatel, Allen Smith, Andrew Rodland, Andy Dougherty, Ben Morrow, brian d foy, Casey West, Chip Salzenberg, Chris 'BinGOs' Williams, Craig A. Berry, David Golden, David Mitchell, Eric Brine, Father Chrysostomos, Florian Ragwitz, George Greer, gregor herrmann, Jan Dubois, Jerry D. Hedden, Jesse Vincent, Joshua Pritikin, Karl Williamson, kmx, Michael G Schwern, Mike Kelly, Nicholas Clark, Paul Green, Rafael Garcia-Suarez, Renee Baecker, Ricardo Signes, Sisyphus, Slaven Rezic, Steffen Müller, Steve Hay, Sullivan Beck, Tatsuhiko Miyagawa, Todd Rinaldo, Tony Cook, Tye McQueen, Vernon Lyon, Walt Mankowski, Zefram, Zsbán Ambrus, Ævar Arnfjörð Bjarmason.

Many of the changes included in this version originated in the CPAN modules included in Perl's core. We're grateful to the entire CPAN community for helping Perl to flourish.

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://rt.perl.org/perlbug/ . 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.

If the bug you are reporting has security implications, which make it inappropriate to send to a publicly archived mailing list, then please send it to perl5-security-report@perl.org. This points to a closed subscription unarchived mailing list, which includes all the core committers, who be able to help

assess the impact of issues, figure out a resolution, and help co-ordinate the release of patches to mitigate or fix the problem across all platforms on which Perl is supported. Please only use this address for security issues in the Perl core, not for modules independently distributed on CPAN.

SEE ALSO

The *Changes* file for an explanation of how to view 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.