NAME

perl5218delta - what is new for perl v5.21.8

DESCRIPTION

This document describes differences between the 5.21.7 release and the 5.21.8 release.

If you are upgrading from an earlier release such as 5.21.6, first read *perl5217delta*, which describes differences between 5.21.6 and 5.21.7.

Notice

With this release we are now in the contentious changes portion of the code freeze as we prepare for the next stable release of Perl 5.

Core Enhancements

The warnings pragma now supports warnings outside of "all"

Ever since perl v5.6.0 we've had no way of adding new warnings without retroactively adding them to all existing programs that used `-w`, `-W` or `use warnings`.

This caused us to not add new useful warnings out of fear that they might unduly burden users who just wanted to upgrade perl and didn't want to deal with a bunch of warnings from their existing code.

We now support a way to have our cake and eat it too, and can add new warnings to the core going forward through other top-level warning categories. See *the warnings documentation* for details.

Non-Capturing Regular Expression Flag

Regular expressions now support a `/n` flag that disables capturing and filling in `$1`, `$2`, etc... inside of groups:

```
"hello" =~ /(hi|hello)/n; # $1 is not set
```

This is equivalent to putting `?:` at the beginning of every capturing group.

See *"n" in perlre* for more information.

prototype with no arguments

`prototype()` with no arguments now infers $_. [perl #123514]

use re 'strict'

This applies stricter syntax rules to regular expression patterns compiled within its scope, which hopefully will alert you to typos and other unintentional behavior that backwards-compatibility issues prevent us from doing in normal regular expression compilations. Because the behavior of this is subject to change in future Perl releases as we gain experience, using this pragma will raise a category `experimental:re_strict` warning. See *'strict' in re*.

New "const" subroutine attribute

The "const" attribute can be applied to an anonymous subroutine. It causes it to be executed immediately when it is cloned. Its value is captured and used to create a new constant subroutine that is returned. This feature is experimental. See *"Constant Functions" in perlsub*.

Incompatible Changes

sub signatures moved before attributes

The experimental sub signatures feature, as introduced in 5.20, parsed signatures after attributes. In this release, the positioning has been moved such that signatures occur in exactly the same positioning a prototype would be found.

Modules and Pragmata

Updated Modules and Pragmata

- *arybase* has been upgraded from version 0.09 to 0.10.
- *attributes* has been upgraded from version 0.24 to 0.25.

  Minor internal change only.
- *autodie* has been upgraded from version 2.25 to 2.26.
- *B* has been upgraded from version 1.54 to 1.55.

  A bug where, after an ithread creation or psuedofork, special/immortal SVs in the child ithread/psuedoprocess did not have the correct class of `B::SPECIAL`, has been fixed.

  The `id` and `outid` PADLIST methods have been added.
- *B::Deparse* has been upgraded from version 1.31 to 1.32.

  Deparsing `BEGIN { undef &foo }` with the **-w** switch enabled started to emit 'uninitialized' warnings in Perl 5.14. This has been fixed.

  Deparsing calls to subs with a `(;+)` prototype resulted in an infinite loop. The `(;$)` `(_)` and `(;_)` prototypes were given the wrong precedence, causing `foo($a<$b)` to be deparsed without the parentheses.
- *Compress::Raw::Bzip2* has been upgraded from version 2.067 to 2.068.
- *Compress::Raw::Zlib* has been upgraded from version 2.067 to 2.068.
- *CPAN::Meta::Requirements* has been upgraded from version 2.130 to 2.131.
- *Data::Dumper* has been upgraded from version 2.155 to 2.156.
- *DB_File* has been upgraded from version 1.834 to 1.835.
- *Devel::Peek* has been upgraded from version 1.20 to 1.21.
- *Devel::PPPort* has been upgraded from version 3.25 to 3.28.
- *Digest::MD5* has been upgraded from version 2.53 to 2.54.
- *Digest::SHA* has been upgraded from version 5.93 to 5.95.
- *DynaLoader* has been upgraded from version 1.29 to 1.30.
- *ExtUtils::Command* has been upgraded from version 1.18 to 1.20.
- *ExtUtils::Manifest* has been upgraded from version 1.69 to 1.70.
- *File::Glob* has been upgraded from version 1.23 to 1.24.

  Avoid SvIV() expanding to call get_sv() three times in a few places. [perl #123606]
- *Filter::Util::Call* has been upgraded from version 1.51 to 1.54.
- *Getopt::Long* has been upgraded from version 2.42 to 2.43.
- *IO::Compress::Base* has been upgraded from version 2.067 to 2.068.
- *IO::Socket::IP* has been upgraded from version 0.34 to 0.36.
- *MIME::Base64* has been upgraded from version 3.14 to 3.15.
- *Module::CoreList* has been upgraded from version 5.20141220 to 5.20150120.
- *Module::Load::Conditional* has been upgraded from version 0.62 to 0.64.

- *Module::Metadata* has been upgraded from version 1.000024 to 1.000026.
- *Opcode* has been upgraded from version 1.30 to 1.31.
- *PerlIO::encoding* has been upgraded from version 0.20 to 0.21.
- *Pod::Simple* has been upgraded from version 3.28 to 3.29.
- *POSIX* has been upgraded from version 1.48 to 1.49.
- *re* has been upgraded from version 0.28 to 0.30.
- *Safe* has been upgraded from version 2.38 to 2.39.

  `reval` was not propagating void context properly.
- *SDBM_File* has been upgraded from version 1.12 to 1.13.

  Simplified the build process. [perl #123413]
- *Test::Harness* has been upgraded from version 3.34 to 3.35.
- *Test::Simple* has been upgraded from version 1.301001_090 to 1.301001_097.
- *Unicode::Collate* has been upgraded from version 1.09 to 1.10.
- *VMS::DCLsym* has been upgraded from version 1.05 to 1.06.
- *warnings* has been upgraded from version 1.29 to 1.30.

Documentation

New Documentation

perlunicook

This document, by Tom Christiansen, provides examples of handling Unicode in Perl.

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*.

New Diagnostics

New Errors

- *Bad symbol for scalar*

  (P) An internal request asked to add a scalar entry to something that wasn't a symbol table entry.
- *:const is not permitted on named subroutines*

  (F) The "const" attribute causes an anonymous subroutine to be run and its value captured at the time that it is cloned. Names subroutines are not cloned like this, so the attribute does not make sense on them.

New Warnings

- *:const is experimental*

  (S experimental::const_attr) The "const" attribute is experimental. If you want to use the feature, disable the warning with `no warnings 'experimental::const_attr'`, but know that in doing so you are taking the risk that your code may break in a future Perl version.
- *Non-finite repeat count does nothing*

  (W numeric) You tried to execute the *x* repetition operator `Inf` (or `-Inf`) or `NaN` times, which doesn't make sense.

- *Useless use of attribute "const"*

  (W misc) The "const" attribute has no effect except on anonymous closure prototypes. You applied it to a subroutine via *attributes.pm*. This is only useful inside an attribute handler for an anonymous subroutine.

- *Unusual use of %s in void context*

  (W void_unusual) Similar to the "Useless use of %s in void context" warning, but only turned on by the top-level "pedantic" warning category, used for e.g. `grep` in void context, which may indicate a bug, but could also just be someone using `grep` for its side-effects as a loop.

  Enabled as part of "extra" warnings, not in the "all" category. See *warnings* for details

- *"use re 'strict'" is experimental*

  (S experimental::re_strict) The things that are different when a regular expression pattern is compiled under `'strict'` are subject to change in future Perl releases in incompatible ways. This means that a pattern that compiles today may not in a future Perl release. This warning is to alert you to that risk.

  Wide character (U+%X) in %s

  (W locale) While in a single-byte locale (*i.e.*, a non-UTF-8 one), a multi-byte character was encountered. Perl considers this character to be the specified Unicode code point. Combining non-UTF8 locales and Unicode is dangerous. Almost certainly some characters will have two different representations. For example, in the ISO 8859-7 (Greek) locale, the code point 0xC3 represents a Capital Gamma. But so also does 0x393. This will make string comparisons unreliable.

  You likely need to figure out how this multi-byte character got mixed up with your single-byte locale (or perhaps you thought you had a UTF-8 locale, but Perl disagrees).

- *Both or neither range ends should be Unicode in regex; marked by <-- HERE in m/%s/*

  (W regexp) (only under `use re 'strict'` or within `(?[...])`)

  In a bracketed character class in a regular expression pattern, you had a range which has exactly one end of it specified using `\N{}`, and the other end is specified using a non-portable mechanism. Perl treats the range as a Unicode range, that is, all the characters in it are considered to be the Unicode characters, and which may be different code points on some platforms Perl runs on. For example, `[\N{U+06}-\x08]` is treated as if you had instead said `[\N{U+06}-\N{U+08}]`, that is it matches the characters whose code points in Unicode are 6, 7, and 8. But that `\x08` might indicate that you meant something different, so the warning gets raised.

- *Ranges of ASCII printables should be some subset of "0-9", "A-Z", or "a-z" in regex; marked by <-- HERE in m/%s/*

  (W regexp) (only under `use re 'strict'` or within `(?[...])`)

  Stricter rules help to find typos and other errors. Perhaps you didn't even intend a range here, if the `"-"` was meant to be some other character, or should have been escaped (like `"\-"`). If you did intend a range, the one that was used is not portable between ASCII and EBCDIC platforms, and doesn't have an obvious meaning to a casual reader.

  ```
  [3-7]    # OK; Obvious and portable
  [d-g]    # OK; Obvious and portable
  [A-Y]    # OK; Obvious and portable
  [A-z]    # WRONG; Not portable; not clear what is meant
  [a-Z]    # WRONG; Not portable; not clear what is meant
  [%-.]    # WRONG; Not portable; not clear what is meant
  [\x41-Z] # WRONG; Not portable; not obvious to non-geek
  ```

  (You can force portability by specifying a Unicode range, which means that the endpoints are

specified by `\N{...}`, but the meaning may still not be obvious.) The stricter rules require that ranges that start or stop with an ASCII character that is not a control have all their endpoints be the literal character, and not some escape sequence (like `"\x41"`), and the ranges must be all digits, or all uppercase letters, or all lowercase letters.

- *Ranges of digits should be from the same group in regex; marked by <-- HERE in m/%s/*

  (W regexp) (only under `use re 'strict'` or within `(?[...])`)

  Stricter rules help to find typos and other errors. You included a range, and at least one of the end points is a decimal digit. Under the stricter rules, when this happens, both end points should be digits in the same group of 10 consecutive digits.

- *"%s" is more clearly written simply as "%s" in regex; marked by <-- HERE in m/%s/*

  (W regexp) (only under `use re 'strict'` or within `(?[...])`)

  You specified a character that has the given plainer way of writing it, and which is also portable to platforms running with different character sets.

Changes to Existing Diagnostics

- The message *Locale '%s' may not work well.%s* is no longer raised unless the problemtatic locale is actually used in the Perl program. Previously it was raised if it merely was the underlying locale. All Perl programs have an underlying locale at all times, but something like a `use locale` is needed for that locale to actually have some effect. This message will not be raised when the underlying locale is hidden.

Configuration and Compilation

- pthreads and lcl will be linked by default if present. This allows XS modules that require threading to work on non-threaded perls. Note that you must still pass `-Dusethreads` if you want a threaded perl.

Testing

- A new test script, *bigmem/subst.t*, has been added to test memory usage of subst on very large strings.
- A new test script, *op/anonconst.t*, has been added to test experimental :const subroutines.
- A new test script, *re/reg_nocapture.t*, has been added to test the new `/n` regexp flag.

Platform Support

Platform-Specific Notes

Win32

- Previously, on Visual C++ for Win64 built Perls only, when compiling every Perl XS module (including CPAN ones) and Perl aware .c file with a 64 bit Visual C++, would unconditionally have around a dozen warnings from hv_func.h. These warnings have been silenced. GCC all bitness and Visual C++ for Win32 were not affected.
- Support for building without PerlIO has been removed from the Windows makefiles. Non-PerlIO builds were all but deprecated in Perl 5.18.0 and are already not supported by *Configure* on POSIX systems.
- Between 2 and 6 ms and 7 I/O calls have been saved per attempt to open a perl module for each path in `@INC`.

Internal Changes

- Added Perl_sv_get_backrefs() to determine if an SV is a weak-referent.

  Function either returns an SV * of type AV, which contains the set of weakreferences which reference the passed in SV, or a simple RV * which is the only weakref to this item.

Selected Bug Fixes

- A bug in regular expression patterns that could lead to segfaults and other crashes has been fixed. This occurred only in patterns compiled with `"/i"`, while taking into account the current POSIX locale (this usually means they have to be compiled within the scope of `"use locale"`), and there must be a string of at least 128 consecutive bytes to match. [perl #123539]
- `s///` now works on very long strings instead of dying with 'Substitution loop'. [perl #103260] [perl #123071]
- `gmtime` no longer crashes with not-a-number values. [perl #123495]
- `\()` (reference to an empty list) and `y///` with lexical `$_` in scope could do a bad write past the end of the stack. They have been fixed to extend the stack first.
- `prototype()` with no arguments used to read the previous item on the stack, so `print "foo", prototype()` would print foo's prototype. It has been fixed to infer `$_` instead. [perl #123514]
- Some cases of lexical state subs inside predeclared subs could crash but no longer do.
- Some cases of nested lexical state subs inside anonymous subs could cause 'Bizarre copy' errors or possibly even crash.
- When trying to emit warnings, perl's default debugger (*perl5db.pl*) was sometimes giving 'Undefined subroutine &DB::db_warn called' instead. This bug, which started to occur in Perl 5.18, has been fixed. [perl #123553]
- Certain syntax errors in substitutions, such as `s/${<>{})//`, would crash, and had done so since Perl 5.10. (In some cases the crash did not start happening till 5.16.) The crash has, of course, been fixed. [perl #123542]
- A repeat expression like `33 x ~3` could cause a large buffer overflow since the new output buffer size was not correctly handled by SvGROW(). An expression like this now properly produces a memory wrap panic. [perl 123554]
- `formline("@...", "a");` would crash. The `FF_CHECKNL` case in pp_formline() didn't set the pointer used to mark the chop position, which led to the `FF_MORE` case crashing with a segmentation fault. This has been fixed. [perl #123538]
- A possible buffer overrun and crash when parsing a literal pattern during regular expression compilation has been fixed. [perl #123604]

Acknowledgements

Perl 5.21.8 represents approximately 4 weeks of development since Perl 5.21.7 and contains approximately 26,000 lines of changes across 750 files from 27 authors.

Excluding auto-generated files, documentation and release tools, there were approximately 13,000 lines of changes to 410 .pm, .t, .c and .h files.

Perl continues to flourish into its third decade thanks to a vibrant community of users and developers. The following people are known to have contributed the improvements that became Perl 5.21.8:

Aaron Crane, Andreas Voegele, Chad Granum, Chris 'BinGOs' Williams, Craig A. Berry, Daniel Dragan, David Mitchell, E. Choroba, Ed J, Father Chrysostomos, H.Merijn Brand, Hugo van der Sanden, James E Keenan, Jarkko Hietaniemi, Karen Etheridge, Karl Williamson, Matthew Horsfall, Max Maischein, Peter Martini, Rafael Garcia-Suarez, Ricardo Signes, Rostislav Skudnov, Slaven Rezic, Steve Hay, Tony Cook, Yves Orton, Ã†var ArnfjÃ¶rÃ° Bjarmason.

The list above is almost certainly incomplete as it is automatically generated from version control history. In particular, it does not include the names of the (very much appreciated) contributors who

reported issues to the Perl bug tracker.

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.

For a more complete list of all of Perl's historical contributors, please see the *AUTHORS* file in the Perl source distribution.

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 https://rt.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.

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 will 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.