NAME

perldelta - what is new for perl v5.24.0

DESCRIPTION

This document describes the differences between the 5.22.0 release and the 5.24.0 release.

Core Enhancements

Postfix dereferencing is no longer experimental

Using the `postderef` and `postderef_qq` features no longer emits a warning. Existing code that disables the `experimental::postderef` warning category that they previously used will continue to work. The `postderef` feature has no effect; all Perl code can use postfix dereferencing, regardless of what feature declarations are in scope. The `5.24` feature bundle now includes the `postderef_qq` feature.

Unicode 8.0 is now supported

For details on what is in this release, see *http://www.unicode.org/versions/Unicode8.0.0/*.

perl will now croak when closing an in-place output file fails

Until now, failure to close the output file for an in-place edit was not detected, meaning that the input file could be clobbered without the edit being successfully completed. Now, when the output file cannot be closed successfully, an exception is raised.

New \b{lb} boundary in regular expressions

`lb` stands for Line Break. It is a Unicode property that determines where a line of text is suitable to break (typically so that it can be output without overflowing the available horizontal space). This capability has long been furnished by the *Unicode::LineBreak* module, but now a light-weight, non-customizable version that is suitable for many purposes is in core Perl.

qr/(?[ ])/ now works in UTF-8 locales

Extended Bracketed Character Classes now will successfully compile when `use locale` is in effect. The compiled pattern will use standard Unicode rules. If the runtime locale is not a UTF-8 one, a warning is raised and standard Unicode rules are used anyway. No tainting is done since the outcome does not actually depend on the locale.

Integer shift (<< and >>) now more explicitly defined

Negative shifts are reverse shifts: left shift becomes right shift, and right shift becomes left shift.

Shifting by the number of bits in a native integer (or more) is zero, except when the "overshift" is right shifting a negative value under `use integer`, in which case the result is -1 (arithmetic shift).

Until now negative shifting and overshifting have been undefined because they have relied on whatever the C implementation happens to do. For example, for the overshift a common C behavior is "modulo shift":

```
1 >> 64 == 1 >> (64 % 64) == 1 >> 0 == 1  # Common C behavior.

# And the same for <<, while Perl now produces 0 for both.
```

Now these behaviors are well-defined under Perl, regardless of what the underlying C implementation does. Note, however, that you are still constrained by the native integer width: you need to know how far left you can go. You can use for example:

```
use Config;
my $wordbits = $Config{uvsize} * 8;  # Or $Config{uvsize} << 3.
```

If you need a more bits on the left shift, you can use for example the `bigint` pragma, or the

`Bit::Vector` module from CPAN.

printf and sprintf now allow reordered precision arguments

That is, `sprintf '|%.*2$d|', 2, 3` now returns `|002|`. This extends the existing reordering mechanism (which allows reordering for arguments that are used as format fields, widths, and vector separators).

More fields provided to sigaction callback with SA_SIGINFO

When passing the `SA_SIGINFO` flag to *sigaction*, the `errno`, `status`, `uid`, `pid`, `addr` and `band` fields are now included in the hash passed to the handler, if supported by the platform.

Hashbang redirection to Perl 6

Previously perl would redirect to another interpreter if it found a hashbang path unless the path contains "perl" (see *perlrun*). To improve compatability with Perl 6 this behavior has been extended to also redirect if "perl" is followed by "6".

Security

Set proper umask before calling mkstemp(3)

In 5.22 perl started setting umask to 0600 before calling `mkstemp(3)` and restoring it afterwards. This wrongfully tells `open(2)` to strip the owner read and write bits from the given mode before applying it, rather than the intended negation of leaving only those bits in place.

Systems that use mode 0666 in `mkstemp(3)` (like old versions of glibc) create a file with permissions 0066, leaving world read and write permissions regardless of current umask.

This has been fixed by using umask 0177 instead. [perl #127322]

Fix out of boundary access in Win32 path handling

This is CVE-2015-8608. For more information see *[perl #126755]*

Fix loss of taint in canonpath

This is CVE-2015-8607. For more information see *[perl #126862]*

Avoid accessing uninitialized memory in win32 crypt()

Added validation that will detect both a short salt and invalid characters in the salt. *[perl #126922]*

Remove duplicate environment variables from environ

Previously, if an environment variable appeared more than once in `environ[]`, `%ENV` would contain the last entry for that name, while a typical `getenv()` would return the first entry. We now make sure `%ENV` contains the same as what `getenv` returns.

Second, we remove duplicates from `environ[]`, so if a setting with that name is set in `%ENV`, we won't pass an unsafe value to a child process.

[CVE-2016-2381]

Incompatible Changes

The autoderef feature has been removed

The experimental `autoderef` feature (which allowed calling `push`, `pop`, `shift`, `unshift`, `splice`, `keys`, `values`, and `each` on a scalar argument) has been deemed unsuccessful. It has now been removed; trying to use the feature (or to disable the `experimental::autoderef` warning it previously triggered) now yields an exception.

Lexical $_ has been removed

`my $_` was introduced in Perl 5.10, and subsequently caused much confusion with no obvious solution. In Perl 5.18.0, it was made experimental on the theory that it would either be removed or redesigned in a less confusing (but backward-incompatible) way. Over the following years, no alternatives were proposed. The feature has now been removed and will fail to compile.

qr/\b{wb}/ is now tailored to Perl expectations

This is now more suited to be a drop-in replacement for plain `\b`, but giving better results for parsing natural language. Previously it strictly followed the current Unicode rules which calls for it to match between each white space character. Now it doesn't generally match within spans of white space, behaving like `\b` does. See *"\b{wb}" in perlrebackslash*

Regular expression compilation errors

Some regular expression patterns that had runtime errors now don't compile at all.

Almost all Unicode properties using the `\p{}` and `\P{}` regular expression pattern constructs are now checked for validity at pattern compilation time, and invalid ones will cause the program to not compile. In earlier releases, this check was often deferred until run time. Whenever an error check is moved from run- to compile time, erroneous code is caught 100% of the time, whereas before it would only get caught if and when the offending portion actually gets executed, which for unreachable code might be never.

qr/\N{}/ now disallowed under use re "strict"

An empty `\N{}` makes no sense, but for backwards compatibility is accepted as doing nothing, though a deprecation warning is raised by default. But now this is a fatal error under the experimental feature *"'strict' mode" in re*.

Nested declarations are now disallowed

A `my`, `our`, or `state` declaration is no longer allowed inside of another `my`, `our`, or `state` declaration.

For example, these are now fatal:

```
   my ($x, my($y));
   our (my $x);
```

[perl #125587]

[perl #121058]

The /\C/ character class has been removed.

This regular expression character class was deprecated in v5.20.0 and has produced a deprecation warning since v5.22.0. It is now a compile-time error. If you need to examine the individual bytes that make up a UTF8-encoded character, then use `utf8::encode()` on the string (or a copy) first.

chdir('') no longer chdirs home

Using `chdir('')` or `chdir(undef)` to chdir home has been deprecated since perl v5.8, and will now fail. Use `chdir()` instead.

ASCII characters in variable names must now be all visible

It was legal until now on ASCII platforms for variable names to contain non-graphical ASCII control characters (ordinals 0 through 31, and 127, which are the C0 controls and `DELETE`). This usage has been deprecated since v5.20, and as of now causes a syntax error. The variables these names referred to are special, reserved by Perl for whatever use it may choose, now, or in the future. Each such variable has an alternative way of spelling it. Instead of the single non-graphic control character, a two character sequence beginning with a caret is used, like `$^]` and `${^GLOBAL_PHASE}`. Details are at *perlvar*. It remains legal, though unwise and deprecated (raising a deprecation warning), to use certain non-graphic non-ASCII characters in variables names when not under `use utf8`. No code should do this, as all such variables are reserved by Perl, and Perl doesn't currently define any of them (but could at any time, without notice).

An off by one issue in $Carp::MaxArgNums has been fixed

`$Carp::MaxArgNums` is supposed to be the number of arguments to display. Prior to this version, it was instead showing `$Carp::MaxArgNums` + 1 arguments, contrary to the documentation.

Only blanks and tabs are now allowed within [...] within (?[...]).

The experimental Extended Bracketed Character Classes can contain regular bracketed character classes within them. These differ from regular ones in that white space is generally ignored, unless escaped by preceding it with a backslash. The white space that is ignored is now limited to just tab `\t` and SPACE characters. Previously, it was any white space. See *"Extended Bracketed Character Classes" in perlrecharclass*.

Deprecations

Using code points above the platform's IV_MAX is now deprecated

Unicode defines code points in the range `0..0x10FFFF`. Some standards at one time defined them up to 2**31 - 1, but Perl has allowed them to be as high as anything that will fit in a word on the platform being used. However, use of those above the platform's `IV_MAX` is broken in some constructs, notably `tr///`, regular expression patterns involving quantifiers, and in some arithmetic and comparison operations, such as being the upper limit of a loop. Now the use of such code points raises a deprecation warning, unless that warning category is turned off. `IV_MAX` is typically 2**31 -1 on 32-bit platforms, and 2**63-1 on 64-bit ones.

Doing bitwise operations on strings containing code points above 0xFF is deprecated

The string bitwise operators treat their operands as strings of bytes, and values beyond 0xFF are nonsensical in this context. To operate on encoded bytes, first encode the strings. To operate on code points' numeric values, use `split` and `map ord`. In the future, this warning will be replaced by an exception.

sysread(), syswrite(), recv() and send() are deprecated on :utf8 handles

The `sysread()`, `recv()`, `syswrite()` and `send()` operators are deprecated on handles that have the `:utf8` layer, either explicitly, or implicitly, eg., with the `:encoding(UTF-16LE)` layer.

Both `sysread()` and `recv()` currently use only the `:utf8` flag for the stream, ignoring the actual layers. Since `sysread()` and `recv()` do no UTF-8 validation they can end up creating invalidly encoded scalars.

Similarly, `syswrite()` and `send()` use only the `:utf8` flag, otherwise ignoring any layers. If the flag is set, both write the value UTF-8 encoded, even if the layer is some different encoding, such as the example above.

Ideally, all of these operators would completely ignore the `:utf8` state, working only with bytes, but this would result in silently breaking existing code. To avoid this a future version of perl will throw an exception when any of `sysread()`, `recv()`, `syswrite()` or `send()` are called on handle with the `:utf8` layer.

Performance Enhancements

- The overhead of scope entry and exit has been considerably reduced, so for example subroutine calls, loops and basic blocks are all faster now. This empty function call now takes about a third less time to execute:

  ```
  sub f{} f();
  ```

- Many languages, such as Chinese, are caseless. Perl now knows about most common ones, and skips much of the work when a program tries to change case in them (like `ucfirst()`) or match caselessly (`qr//i`). This will speed up a program, such as a web server, that can operate on multiple languages, while it is operating on a caseless one.
- `/fixed-substr/` has been made much faster.

On platforms with a libc `memchr()` implementation which makes good use of underlying hardware support, patterns which include fixed substrings will now often be much faster; for example with glibc on a recent x86_64 CPU, this:

```
    $s = "a" x 1000 . "wxyz";
    $s =~ /wxyz/ for 1..30000
```

is now about 7 times faster. On systems with slow `memchr()`, e.g. 32-bit ARM Raspberry Pi, there will be a small or little speedup. Conversely, some pathological cases, such as `"ab" x 1000 =~ /aa/` will be slower now; up to 3 times slower on the rPi, 1.5x slower on x86_64.

- Faster addition, subtraction and multiplication.

  Since 5.8.0, arithmetic became slower due to the need to support 64-bit integers. To deal with 64-bit integers, a lot more corner cases need to be checked, which adds time. We now detect common cases where there is no need to check for those corner cases, and special-case them.

- Preincrement, predecrement, postincrement, and postdecrement have been made faster by internally splitting the functions which handled multiple cases into different functions.

- Creating Perl debugger data structures (see *"Debugger Internals" in perldebguts*) for XSUBs and const subs has been removed. This removed one glob/scalar combo for each unique `.c` file that XSUBs and const subs came from. On startup (`perl -e"0"`) about half a dozen glob/scalar debugger combos were created. Loading XS modules created more glob/scalar combos. These things were being created regardless of whether the perl debugger was being used, and despite the fact that it can't debug C code anyway

- On Win32, `stat`ing or `-X`ing a path, if the file or directory does not exist, is now 3.5x faster than before.

- Single arguments in list assign are now slightly faster:

```
  ($x) = (...);
  (...) = ($x);
```

- Less peak memory is now used when compiling regular expression patterns.

Modules and Pragmata

Updated Modules and Pragmata

- *arybase* has been upgraded from version 0.10 to 0.11.
- *Attribute::Handlers* has been upgraded from version 0.97 to 0.99.
- *autodie* has been upgraded from version 2.26 to 2.29.
- *autouse* has been upgraded from version 1.08 to 1.11.
- *B* has been upgraded from version 1.58 to 1.62.
- *B::Deparse* has been upgraded from version 1.35 to 1.37.
- *base* has been upgraded from version 2.22 to 2.23.
- *Benchmark* has been upgraded from version 1.2 to 1.22.
- *bignum* has been upgraded from version 0.39 to 0.42.
- *bytes* has been upgraded from version 1.04 to 1.05.
- *Carp* has been upgraded from version 1.36 to 1.40.
- *Compress::Raw::Bzip2* has been upgraded from version 2.068 to 2.069.

- *Compress::Raw::Zlib* has been upgraded from version 2.068 to 2.069.
- *Config::Perl::V* has been upgraded from version 0.24 to 0.25.
- *CPAN::Meta* has been upgraded from version 2.150001 to 2.150005.
- *CPAN::Meta::Requirements* has been upgraded from version 2.132 to 2.140.
- *CPAN::Meta::YAML* has been upgraded from version 0.012 to 0.018.
- *Data::Dumper* has been upgraded from version 2.158 to 2.160.
- *Devel::Peek* has been upgraded from version 1.22 to 1.23.
- *Devel::PPPort* has been upgraded from version 3.31 to 3.32.
- *Dumpvalue* has been upgraded from version 1.17 to 1.18.
- *DynaLoader* has been upgraded from version 1.32 to 1.38.
- *Encode* has been upgraded from version 2.72 to 2.80.
- *encoding* has been upgraded from version 2.14 to 2.17.
- *encoding::warnings* has been upgraded from version 0.11 to 0.12.
- *English* has been upgraded from version 1.09 to 1.10.
- *Errno* has been upgraded from version 1.23 to 1.25.
- *experimental* has been upgraded from version 0.013 to 0.016.
- *ExtUtils::CBuilder* has been upgraded from version 0.280221 to 0.280225.
- *ExtUtils::Embed* has been upgraded from version 1.32 to 1.33.
- *ExtUtils::MakeMaker* has been upgraded from version 7.04_01 to 7.10_01.
- *ExtUtils::ParseXS* has been upgraded from version 3.28 to 3.31.
- *ExtUtils::Typemaps* has been upgraded from version 3.28 to 3.31.
- *feature* has been upgraded from version 1.40 to 1.42.
- *fields* has been upgraded from version 2.17 to 2.23.
- *File::Copy* has been upgraded from version 2.30 to 2.31.
- *File::Find* has been upgraded from version 1.29 to 1.34.
- *File::Glob* has been upgraded from version 1.24 to 1.26.
- *File::Path* has been upgraded from version 2.09 to 2.12_01.
- *File::Spec* has been upgraded from version 3.56 to 3.63.
- *Filter::Util::Call* has been upgraded from version 1.54 to 1.55.
- *Getopt::Long* has been upgraded from version 2.45 to 2.48.
- *Hash::Util* has been upgraded from version 0.18 to 0.19.
- *Hash::Util::FieldHash* has been upgraded from version 1.15 to 1.19.
- *HTTP::Tiny* has been upgraded from version 0.054 to 0.056.
- *I18N::Langinfo* has been upgraded from version 0.12 to 0.13.

- *if* has been upgraded from version 0.0604 to 0.0606.
- *IO* has been upgraded from version 1.35 to 1.36.
- IO-Compress has been upgraded from version 2.068 to 2.069.
- *IPC::Open3* has been upgraded from version 1.18 to 1.20.
- *IPC::SysV* has been upgraded from version 2.04 to 2.06_01.
- *List::Util* has been upgraded from version 1.41 to 1.42_02.
- *locale* has been upgraded from version 1.06 to 1.08.
- *Locale::Codes* has been upgraded from version 3.34 to 3.37.
- *Math::BigInt* has been upgraded from version 1.9997 to 1.999715.
- *Math::BigInt::FastCalc* has been upgraded from version 0.31 to 0.40.
- *Math::BigRat* has been upgraded from version 0.2608 to 0.260802.
- *Module::CoreList* has been upgraded from version 5.20150520 to 5.20160506.
- *Module::Metadata* has been upgraded from version 1.000026 to 1.000031.
- *mro* has been upgraded from version 1.17 to 1.18.
- *ODBM_File* has been upgraded from version 1.12 to 1.14.
- *Opcode* has been upgraded from version 1.32 to 1.34.
- *parent* has been upgraded from version 0.232 to 0.234.
- *Parse::CPAN::Meta* has been upgraded from version 1.4414 to 1.4417.
- *Perl::OSType* has been upgraded from version 1.008 to 1.009.
- *perlfaq* has been upgraded from version 5.021009 to 5.021010.
- *PerlIO::encoding* has been upgraded from version 0.21 to 0.24.
- *PerlIO::mmap* has been upgraded from version 0.014 to 0.016.
- *PerlIO::scalar* has been upgraded from version 0.22 to 0.24.
- *PerlIO::via* has been upgraded from version 0.15 to 0.16.
- podlators has been upgraded from version 2.28 to 4.07.
- *Pod::Functions* has been upgraded from version 1.09 to 1.10.
- *Pod::Perldoc* has been upgraded from version 3.25 to 3.25_02.
- *Pod::Simple* has been upgraded from version 3.29 to 3.32.
- *Pod::Usage* has been upgraded from version 1.64 to 1.68.
- *POSIX* has been upgraded from version 1.53 to 1.65.
- *Scalar::Util* has been upgraded from version 1.41 to 1.42_02.
- *SDBM_File* has been upgraded from version 1.13 to 1.14.
- *SelfLoader* has been upgraded from version 1.22 to 1.23.
- *Socket* has been upgraded from version 2.018 to 2.020_03.

- *Storable* has been upgraded from version 2.53 to 2.56.
- *strict* has been upgraded from version 1.09 to 1.11.
- *Term::ANSIColor* has been upgraded from version 4.03 to 4.04.
- *Term::Cap* has been upgraded from version 1.15 to 1.17.
- *Test* has been upgraded from version 1.26 to 1.28.
- *Test::Harness* has been upgraded from version 3.35 to 3.36.
- *Thread::Queue* has been upgraded from version 3.05 to 3.09.
- *threads* has been upgraded from version 2.01 to 2.07.
- *threads::shared* has been upgraded from version 1.48 to 1.51.
- *Tie::File* has been upgraded from version 1.01 to 1.02.
- *Tie::Scalar* has been upgraded from version 1.03 to 1.04.
- *Time::HiRes* has been upgraded from version 1.9726 to 1.9733.
- *Time::Piece* has been upgraded from version 1.29 to 1.31.
- *Unicode::Collate* has been upgraded from version 1.12 to 1.14.
- *Unicode::Normalize* has been upgraded from version 1.18 to 1.25.
- *Unicode::UCD* has been upgraded from version 0.61 to 0.64.
- *UNIVERSAL* has been upgraded from version 1.12 to 1.13.
- *utf8* has been upgraded from version 1.17 to 1.19.
- *version* has been upgraded from version 0.9909 to 0.9916.
- *warnings* has been upgraded from version 1.32 to 1.36.
- *Win32* has been upgraded from version 0.51 to 0.52.
- *Win32API::File* has been upgraded from version 0.1202 to 0.1203.
- *XS::Typemap* has been upgraded from version 0.13 to 0.14.
- *XSLoader* has been upgraded from version 0.20 to 0.21.

Documentation

Changes to Existing Documentation

perlapi

- The process of using undocumented globals has been documented, namely, that one should send email to *perl5-porters@perl.org* first to get the go-ahead for documenting and using an undocumented function or global variable.

perlcall

- A number of cleanups have been made to perlcall, including:
  - use `EXTEND(SP, n)` and `PUSHs()` instead of `XPUSHs()` where applicable and update prose to match
  - add POPu, POPul and POPpbytex to the "complete list of POP macros" and clarify the documentation for some of the existing entries, and a note about side-effects

- add API documentation for POPu and POPul
- use ERRSV more efficiently
- approaches to thread-safety storage of SVs.

perlfunc

- The documentation of `hex` has been revised to clarify valid inputs.
- Better explain meaning of negative PIDs in `waitpid`. *[perl #127080]*
- General cleanup: there's more consistency now (in POD usage, grammar, code examples), better practices in code examples (use of `my`, removal of bareword filehandles, dropped usage of `&` when calling subroutines, ...), etc.

perlguts

- A new section has been added, *"Dynamic Scope and the Context Stack" in perlguts*, which explains how the perl context stack works.

perllocale

- A stronger caution about using locales in threaded applications is given. Locales are not thread-safe, and you can get wrong results or even segfaults if you use them there.

perlmodlib

- We now recommend contacting the module-authors list or PAUSE in seeking guidance on the naming of modules.

perlop

- The documentation of `qx//` now describes how `$?` is affected.

perlpolicy

- This note has been added to perlpolicy:

  ```
  While civility is required, kindness is encouraged; if you have any
  doubt about whether you are being civil, simply ask yourself, "Am I
  being kind?" and aspire to that.
  ```

perlreftut

- Fix some examples to be *strict* clean.

perlrebackslash

- Clarify that in languages like Japanese and Thai, dictionary lookup is required to determine word boundaries.

perlsub

- Updated to note that anonymous subroutines can have signatures.

perlsyn

- Fixed a broken example where `=` was used instead of `==` in conditional in do/while example.

perltie

- The usage of `FIRSTKEY` and `NEXTKEY` has been clarified.

perlunicode

- Discourage use of 'In' as a prefix signifying the Unicode Block property.

perlvar

- The documentation of `$@` was reworded to clarify that it is not just for syntax errors in `eval`. *[perl #124034]*
- The specific true value of `$!{E...}` is now documented, noting that it is subject to change and not guaranteed.
- Use of `$OLD_PERL_VERSION` is now discouraged.

perlxs

- The documentation of `PROTOTYPES` has been corrected; they are *disabled* by default, not *enabled*.

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

- *%s must not be a named sequence in transliteration operator*
- *Can't find Unicode property definition "%s" in regex;*
- *Can't redeclare "%s" in "%s"*
- *Character following \p must be '{' or a single-character Unicode property name in regex;*
- *Empty \%c in regex; marked by <-- HERE in m/%s/*
- *Illegal user-defined property name*
- *Invalid number '%s' for -C option.*
- *Sequence (?... not terminated in regex; marked by <-- HERE in m/%s/*
- *Sequence (?P<... not terminated in regex; marked by <-- HERE in m/%s/*
- *Sequence (?P>... not terminated in regex; marked by <-- HERE in m/%s/*

New Warnings

- *Assuming NOT a POSIX class since %s in regex; marked by <-- HERE in m/%s/*
- *%s() is deprecated on :utf8 handles*

Changes to Existing Diagnostics

- Accessing the `IO` part of a glob as `FILEHANDLE` instead of `IO` is no longer deprecated. It is discouraged to encourage uniformity (so that, for example, one can grep more easily) but it will not be removed. *[perl #127060]*
- The diagnostic `Hexadecimal float: internal error` has been changed to `Hexadecimal float: internal error (%s)` to include more information.
- *Can't modify non-lvalue subroutine call of &%s*

  This error now reports the name of the non-lvalue subroutine you attempted to use as an lvalue.
- When running out of memory during an attempt the increase the stack size, previously, perl would die using the cryptic message `panic: av_extend_guts() negative count (-9223372036854775681)`. This has been fixed to show the prettier message: *Out of memory during stack extend*

Configuration and Compilation

- `Configure` now acts as if the `-O` option is always passed, allowing command line options to override saved configuration. This should eliminate confusion when command line options are ignored for no obvious reason. `-O` is now permitted, but ignored.
- Bison 3.0 is now supported.
- *Configure* no longer probes for *libnm* by default. Originally this was the "New Math" library, but the name has been re-used by the GNOME NetworkManager. *[perl #127131]*
- Added *Configure* probes for `newlocale`, `freelocale`, and `uselocale`.
- `PPPort.so/PPPort.dll` no longer get installed, as they are not used by `PPPort.pm`, only by its test files.
- It is now possible to specify which compilation date to show on `perl -V` output, by setting the macro `PERL_BUILD_DATE`.
- Using the `NO_HASH_SEED` define in combination with the default hash algorithm `PERL_HASH_FUNC_ONE_AT_A_TIME_HARD` resulted in a fatal error while compiling the interpreter, since Perl 5.17.10. This has been fixed.
- *Configure* should handle spaces in paths a little better.
- No longer generate EBCDIC POSIX-BC tables. We don't believe anyone is using Perl and POSIX-BC at this time, and by not generating these tables it saves time during development, and makes the resulting tar ball smaller.
- The GNU Make makefile for Win32 now supports parallel builds. [perl #126632]
- You can now build perl with MSVC++ on Win32 using GNU Make. [perl #126632]
- The Win32 miniperl now has a real `getcwd` which increases build performance resulting in `getcwd()` being 605x faster in Win32 miniperl.
- Configure now takes `-Dusequadmath` into account when calculating the `alignbytes` configuration variable. Previously the mis-calculated `alignbytes` could cause alignment errors on debugging builds. [perl #127894]

Testing

- A new test (*t/op/aassign.t*) has been added to test the list assignment operator `OP_AASSIGN`.
- Parallel building has been added to the dmake `makefile.mk` makefile. All Win32 compilers are supported.

Platform Support

Platform-Specific Notes

AmigaOS

- The AmigaOS port has been reintegrated into the main tree, based off of Perl 5.22.1.

Cygwin

- Tests are more robust against unusual cygdrive prefixes. *[perl #126834]*

EBCDIC

UTF-EBCDIC extended

UTF-EBCDIC is like UTF-8, but for EBCDIC platforms. It now has been extended so that it can represent code points up to 2 ** 64 - 1 on platforms with 64-bit words. This brings it into parity with UTF-8. This enhancement requires an incompatible change to the representation of code points in the range 2 ** 30 to 2 ** 31 -1 (the latter was the previous maximum representable code point). This means that a file that contains one of these code points, written out with previous versions of perl cannot be read in, without conversion, by a perl containing this change. We do not believe any such files are in existence, but if you do have one, submit a ticket at *perlbug@perl.org*, and we will write a conversion script for you.

EBCDIC `cmp()` and `sort()` fixed for UTF-EBCDIC strings

Comparing two strings that were both encoded in UTF-8 (or more precisely, UTF-EBCDIC) did not work properly until now. Since `sort()` uses `cmp()`, this fixes that as well.

EBCDIC `tr///` and `y///` fixed for `\N{}`, and `use utf8` ranges

Perl v5.22 introduced the concept of portable ranges to regular expression patterns. A portable range matches the same set of characters no matter what platform is being run on. This concept is now extended to `tr///`. See *`tr///`*.

There were also some problems with these operations under `use utf8`, which are now fixed

FreeBSD

- Use the `fdclose()` function from FreeBSD if it is available. *[perl #126847]*

IRIX

- Under some circumstances IRIX stdio `fgetc()` and `fread()` set the errno to `ENOENT`, which made no sense according to either IRIX or POSIX docs. Errno is now cleared in such cases. *[perl #123977]*
- Problems when multiplying long doubles by infinity have been fixed. *[perl #126396]*

MacOS X

- Until now OS X builds of perl have specified a link target of 10.3 (Panther, 2003) but have not specified a compiler target. From now on, builds of perl on OS X 10.6 or later (Snow Leopard, 2008) by default capture the current OS X version and specify that as the explicit build target in both compiler and linker flags, thus preserving binary compatibility for extensions built later regardless of changes in OS X, SDK, or compiler and linker versions. To override the default value used in the build and preserved in the flags, specify `export MACOSX_DEPLOYMENT_TARGET=10.N` before configuring and building perl, where 10.N is the version of OS X you wish to target. In OS X 10.5 or earlier there is no change to the behavior present when those systems were current; the link target is still OS X 10.3 and there is no explicit compiler target.
- Builds with both -DDEBUGGING and threading enabled would fail with a "panic: free from wrong pool" error when built or tested from Terminal on OS X. This was caused by perl's internal management of the environment conflicting with an atfork handler using the libc `setenv()` function to update the environment.

  Perl now uses `setenv()`/`unsetenv()` to update the environment on OS X. *[perl #126240]*

Solaris

- All Solaris variants now build a shared libperl

  Solaris and variants like OpenIndiana now always build with the shared Perl library (Configure -Duseshrplib). This was required for the OpenIndiana builds, but this has also been the setting for Oracle/Sun Perl builds for several years.

Tru64

- Workaround where Tru64 balks when prototypes are listed as `PERL_STATIC_INLINE`, but where the test is build with `-DPERL_NO_INLINE_FUNCTIONS`.

VMS

- On VMS, the math function prototypes in `math.h` are now visible under C++. Now building the POSIX extension with C++ will no longer crash.
- VMS has had `setenv`/`unsetenv` since v7.0 (released in 1996), `Perl_vmssetenv` now always uses `setenv`/`unsetenv`.
- Perl now implements its own `killpg` by scanning for processes in the specified process group, which may not mean exactly the same thing as a Unix process group, but allows us to send a signal to a parent (or master) process and all of its sub-processes. At the perl level, this means we can now send a negative pid like so:

  ```
  kill SIGKILL, -$pid;
  ```

  to signal all processes in the same group as `$pid`.
- For those `%ENV` elements based on the CRTL environ array, we've always preserved case when setting them but did look-ups only after upcasing the key first, which made lower- or mixed-case entries go missing. This problem has been corrected by making `%ENV` elements derived from the environ array case-sensitive on look-up as well as case-preserving on store.
- Environment look-ups for `PERL5LIB` and `PERLLIB` previously only considered logical names, but now consider all sources of `%ENV` as determined by `PERL_ENV_TABLES` and as documented in *"%ENV" in perlvms*.
- The minimum supported version of VMS is now v7.3-2, released in 2003. As a side effect of this change, VAX is no longer supported as the terminal release of OpenVMS VAX was v7.3 in 2001.

Win32

- A new build option `USE_NO_REGISTRY` has been added to the makefiles. This option is off by default, meaning the default is to do Windows registry lookups. This option stops Perl from looking inside the registry for anything. For what values are looked up in the registry see *perlwin32*. Internally, in C, the name of this option is `WIN32_NO_REGISTRY`.
- The behavior of Perl using `HKEY_CURRENT_USER\Software\Perl` and `HKEY_LOCAL_MACHINE\Software\Perl` to lookup certain values, including `%ENV` vars starting with `PERL` has changed. Previously, the 2 keys were checked for entries at all times through the perl process's life time even if they did not exist. For performance reasons, now, if the root key (i.e. `HKEY_CURRENT_USER\Software\Perl` or `HKEY_LOCAL_MACHINE\Software\Perl`) does not exist at process start time, it will not be checked again for `%ENV` override entries for the remainder of the perl process's life. This more closely matches Unix behavior in that the environment is copied or inherited on startup and changing the variable in the parent process or another process or editing *.bashrc* will not change the environmental variable in other existing, running, processes.
- One glob fetch was removed for each `-X` or `stat` call whether done from Perl code or internally from Perl's C code. The glob being looked up was `${^WIN32_SLOPPY_STAT}` which is a special variable. This makes `-X` and `stat` slightly faster.
- During miniperl's process startup, during the build process, 4 to 8 IO calls related to the process starting *.pl* and the *buildcustomize.pl* file were removed from the code opening and executing the first 1 or 2 *.pl* files.

- Builds using Microsoft Visual C++ 2003 and earlier no longer produce an "INTERNAL COMPILER ERROR" message. [perl #126045]
- Visual C++ 2013 builds will now execute on XP and higher. Previously they would only execute on Vista and higher.
- You can now build perl with GNU Make and GCC. [perl #123440]
- `truncate($filename, $size)` now works for files over 4GB in size. [perl #125347]
- Parallel building has been added to the dmake `makefile.mk` makefile. All Win32 compilers are supported.
- Building a 64-bit perl with a 64-bit GCC but a 32-bit gmake would result in an invalid `$Config{archname}` for the resulting perl. [perl #127584]
- Errors set by Winsock functions are now put directly into `$^E`, and the relevant `WSAE*` error codes are now exported from the *Errno* and *POSIX* modules for testing this against.

  The previous behavior of putting the errors (converted to POSIX-style `E*` error codes since Perl 5.20.0) into `$!` was buggy due to the non-equivalence of like-named Winsock and POSIX error constants, a relationship between which has unfortunately been established in one way or another since Perl 5.8.0.

  The new behavior provides a much more robust solution for checking Winsock errors in portable software without accidentally matching POSIX tests that were intended for other OSes and may have different meanings for Winsock.

  The old behavior is currently retained, warts and all, for backwards compatibility, but users are encouraged to change any code that tests `$!` against `E*` constants for Winsock errors to instead test `$^E` against `WSAE*` constants. After a suitable deprecation period, the old behavior may be removed, leaving `$!` unchanged after Winsock function calls, to avoid any possible confusion over which error variable to check.

ppc64el

floating point

The floating point format of ppc64el (Debian naming for little-endian PowerPC) is now detected correctly.

Internal Changes

- The implementation of perl's context stack system, and its internal API, have been heavily reworked. Note that no significant changes have been made to any external APIs, but XS code which relies on such internal details may need to be fixed. The main changes are:
  - The `PUSHBLOCK()`, `POPSUB()` etc. macros have been replaced with static inline functions such as `cx_pushblock()`, `cx_popsub()` etc. These use function args rather than implicitly relying on local vars such as `gimme` and `newsp` being available. Also their functionality has changed: in particular, `cx_popblock()` no longer decrements `cxstack_ix`. The ordering of the steps in the `pp_leave*` functions involving `cx_popblock()`, `cx_popsub()` etc. has changed. See the new documentation, *"Dynamic Scope and the Context Stack" in perlguts*, for details on how to use them.
  - Various macros, which now consistently have a CX_ prefix, have been added:

    ```
    CX_CUR(), CX_LEAVE_SCOPE(), CX_POP()
    ```

    or renamed:

```
CX_POP_SAVEARRAY(), CX_DEBUG(), CX_PUSHSUBST(), CX_POPSUBST()
```

  - `cx_pushblock()` now saves `PL_savestack_ix` and `PL_tmps_floor`, so `pp_enter*` and `pp_leave*` no longer do

    ```
    ENTER; SAVETMPS; ....; LEAVE
    ```

  - `cx_popblock()` now also restores `PL_curpm`.
  - In `dounwind()` for every context type, the current savestack frame is now processed before each context is popped; formerly this was only done for sub-like context frames. This action has been removed from `cx_popsub()` and placed into its own macro, `CX_LEAVE_SCOPE(cx)`, which must be called before `cx_popsub()` etc.

    `dounwind()` now also does a `cx_popblock()` on the last popped frame (formerly it only did the `cx_popsub()` etc. actions on each frame).
  - The temps stack is now freed on scope exit; previously, temps created during the last statement of a block wouldn't be freed until the next `nextstate` following the block (apart from an existing hack that did this for recursive subs in scalar context); and in something like `f(g())`, the temps created by the last statement in `g()` would formerly not be freed until the statement following the return from `f()`.
  - Most values that were saved on the savestack on scope entry are now saved in suitable new fields in the context struct, and saved and restored directly by `cx_pushfoo()` and `cx_popfoo()`, which is much faster.
  - Various context struct fields have been added, removed or modified.
  - The handling of `@_` in `cx_pushsub()` and `cx_popsub()` has been considerably tidied up, including removing the `argarray` field from the context struct, and extracting out some common (but rarely used) code into a separate function, `clear_defarray()`. Also, useful subsets of `cx_popsub()` which had been unrolled in places like `pp_goto` have been gathered into the new functions `cx_popsub_args()` and `cx_popsub_common()`.
  - `pp_leavesub` and `pp_leavesublv` now use the same function as the rest of the `pp_leave*`'s to process return args.
  - `CXp_FOR_PAD` and `CXp_FOR_GV` flags have been added, and `CXt_LOOP_FOR` has been split into `CXt_LOOP_LIST`, `CXt_LOOP_ARY`.
  - Some variables formerly declared by `dMULTICALL` (but not documented) have been removed.

- The obscure `PL_timesbuf` variable, effectively a vestige of Perl 1, has been removed. It was documented as deprecated in Perl 5.20, with a statement that it would be removed early in the 5.21.x series; that has now finally happened. *[perl #121351]*
- An unwarranted assertion in `Perl_newATTRSUB_x()` has been removed. If a stub subroutine definition with a prototype has been seen, then any subsequent stub (or definition) of the same subroutine with an attribute was causing an assertion failure because of a null pointer. *[perl #126845]*
- `::` has been replaced by `__` in `ExtUtils::ParseXS`, like it's done for parameters/return values. This is more consistent, and simplifies writing XS code wrapping C++ classes into a nested Perl namespace (it requires only a typedef for `Foo__Bar` rather than two, one for `Foo_Bar` and the other for `Foo::Bar`).
- The `to_utf8_case()` function is now deprecated. Instead use `toUPPER_utf8`,

toTITLE_utf8, toLOWER_utf8, and toFOLD_utf8. (See *http://nntp.perl.org/group/perl.perl5.porters/233287*.)

- Perl core code and the threads extension have been annotated so that, if Perl is configured to use threads, then during compile-time clang (3.6 or later) will warn about suspicious uses of mutexes. See *http://clang.llvm.org/docs/ThreadSafetyAnalysis.html* for more information.
- The `signbit()` emulation has been enhanced. This will help older and/or more exotic platforms or configurations.
- Most EBCDIC-specific code in the core has been unified with non-EBCDIC code, to avoid repetition and make maintenance easier.
- MSWin32 code for `$^X` has been moved out of the *win32* directory to *caretx.c*, where other operating systems set that variable.
- `sv_ref()` is now part of the API.
- *"sv_backoff" in perlapi* had its return type changed from `int` to `void`. It previously has always returned `0` since Perl 5.000 stable but that was undocumented. Although `sv_backoff` is marked as public API, XS code is not expected to be impacted since the proper API call would be through public API `sv_setsv(sv, &PL_sv_undef)`, or quasi-public `SvOOK_off`, or non-public `SvOK_off` calls, and the return value of `sv_backoff` was previously a meaningless constant that can be rewritten as `(sv_backoff(sv),0)`.
- The `EXTEND` and `MEXTEND` macros have been improved to avoid various issues with integer truncation and wrapping. In particular, some casts formerly used within the macros have been removed. This means for example that passing an unsigned `nitems` argument is likely to raise a compiler warning now (it's always been documented to require a signed value; formerly int, lately SSize_t).
- `PL_sawalias` and `GPf_ALIASED_SV` have been removed.
- `GvASSIGN_GENERATION` and `GvASSIGN_GENERATION_set` have been removed.

Selected Bug Fixes

- It now works properly to specify a user-defined property, such as

  ```
  qr/\p{mypkg1::IsMyProperty}/i
  ```

  with `/i` caseless matching, an explicit package name, and *IsMyProperty* not defined at the time of the pattern compilation.
- Perl's `memcpy()`, `memmove()`, `memset()` and `memcmp()` fallbacks are now more compatible with the originals. [perl #127619]
- Fixed the issue where a `s///r`) with **-DPERL_NO_COW** attempts to modify the source SV, resulting in the program dying. [perl #127635]
- Fixed an EBCDIC-platform-only case where a pattern could fail to match. This occurred when matching characters from the set of C1 controls when the target matched string was in UTF-8.
- Narrow the filename check in *strict.pm* and *warnings.pm*. Previously, it assumed that if the filename (without the *.pmc?* extension) differed from the package name, if was a misspelled use statement (i.e. `use Strict` instead of `use strict`). We now check whether there's really a miscapitalization happening, and not some other issue.
- Turn an assertion into a more user friendly failure when parsing regexes. [perl #127599]
- Correctly raise an error when trying to compile patterns with unterminated character classes while there are trailing backslashes. [perl #126141].

- Line numbers larger than 2**31-1 but less than 2**32 are no longer returned by `caller()` as negative numbers. [perl #126991]
- `unless ( *assignment* )` now properly warns when syntax warnings are enabled. [perl #127122]
- Setting an `ISA` glob to an array reference now properly adds `isaelem` magic to any existing elements. Previously modifying such an element would not update the ISA cache, so method calls would call the wrong function. Perl would also crash if the `ISA` glob was destroyed, since new code added in 5.23.7 would try to release the `isaelem` magic from the elements. [perl #127351]
- If a here-doc was found while parsing another operator, the parser had already read end of file, and the here-doc was not terminated, perl could produce an assertion or a segmentation fault. This now reliably complains about the unterminated here-doc. [perl #125540]
- `untie()` would sometimes return the last value returned by the `UNTIE()` handler as well as it's normal value, messing up the stack. [perl #126621]
- Fixed an operator precedence problem when `castflags & 2` is true. [perl #127474]
- Caching of DESTROY methods could result in a non-pointer or a non-STASH stored in the `SvSTASH()` slot of a stash, breaking the B `STASH()` method. The DESTROY method is now cached in the MRO metadata for the stash. [perl #126410]
- The AUTOLOAD method is now called when searching for a DESTROY method, and correctly sets `$AUTOLOAD` too. [perl #124387] [perl #127494]
- Avoid parsing beyond the end of the buffer when processing a `#line` directive with no filename. [perl #127334]
- Perl now raises a warning when a regular expression pattern looks like it was supposed to contain a POSIX class, like `qr/[[:alpha:]]/`, but there was some slight defect in its specification which causes it to instead be treated as a regular bracketed character class. An example would be missing the second colon in the above like this: `qr/[[:alpha]]/`. This compiles to match a sequence of two characters. The second is `"]"`, and the first is any of: `"["`, `":"`, `"a"`, `"h"`, `"l"`, or `"p"`. This is unlikely to be the intended meaning, and now a warning is raised. No warning is raised unless the specification is very close to one of the 14 legal POSIX classes. (See *"POSIX Character Classes" in perlrecharclass*.) [perl #8904]
- Certain regex patterns involving a complemented POSIX class in an inverted bracketed character class, and matching something else optionally would improperly fail to match. An example of one that could fail is `qr/_?[^\Wbar]\x{100}/`. This has been fixed. [perl #127537]
- Perl 5.22 added support to the C99 hexadecimal floating point notation, but sometimes misparses hex floats. This has been fixed. [perl #127183]
- A regression that allowed undeclared barewords in hash keys to work despite strictures has been fixed. *[perl #126981]*
- Calls to the placeholder `&PL_sv_yes` used internally when an `import()` or `unimport()` method isn't found now correctly handle scalar context. *[perl #126042]*
- Report more context when we see an array where we expect to see an operator and avoid an assertion failure. *[perl #123737]*
- Modifying an array that was previously a package `@ISA` no longer causes assertion failures or crashes. *[perl #123788]*
- Retain binary compatibility across plain and DEBUGGING perl builds. *[perl #127212]*

- Avoid leaking memory when setting `$ENV{foo}` on darwin. *[perl #126240]*
- `/...\G/` no longer crashes on utf8 strings. When `\G` is a fixed number of characters from the start of the regex, perl needs to count back that many characters from the current `pos()` position and start matching from there. However, it was counting back bytes rather than characters, which could lead to panics on utf8 strings.
- In some cases operators that return integers would return negative integers as large positive integers. *[perl #126635]*
- The `pipe()` operator would assert for DEBUGGING builds instead of producing the correct error message. The condition asserted on is detected and reported on correctly without the assertions, so the assertions were removed. *[perl #126480]*
- In some cases, failing to parse a here-doc would attempt to use freed memory. This was caused by a pointer not being restored correctly. *[perl #126443]*
- `@x = sort { *a = 0; $a <=> $b } 0 .. 1` no longer frees the GP for *a before restoring its SV slot. *[perl #124097]*
- Multiple problems with the new hexadecimal floating point printf format `%a` were fixed: *[perl #126582]*, *[perl #126586]*, *[perl #126822]*
- Calling `mg_set()` in `leave_scope()` no longer leaks.
- A regression from Perl v5.20 was fixed in which debugging output of regular expression compilation was wrong. (The pattern was correctly compiled, but what got displayed for it was wrong.)
- `\b{sb}` works much better. In Perl v5.22.0, this new construct didn't seem to give the expected results, yet passed all the tests in the extensive suite furnished by Unicode. It turns out that it was because these were short input strings, and the failures had to do with longer inputs.
- Certain syntax errors in *"Extended Bracketed Character Classes" in perlrecharclass* caused panics instead of the proper error message. This has now been fixed. [perl #126481]
- Perl 5.20 added a message when a quantifier in a regular expression was useless, but then caused the parser to skip it; this caused the surplus quantifier to be silently ignored, instead of throwing an error. This is now fixed. [perl #126253]
- The switch to building non-XS modules last in win32/makefile.mk (introduced by design as part of the changes to enable parallel building) caused the build of POSIX to break due to problems with the version module. This is now fixed.
- Improved parsing of hex float constants.
- Fixed an issue with `pack` where `pack "H"` (and `pack "h"`) could read past the source when given a non-utf8 source, and a utf8 target. [perl #126325]
- Fixed several cases where perl would abort due to a segmentation fault, or a C-level assert. [perl #126615], [perl #126602], [perl #126193].
- There were places in regular expression patterns where comments (`(?#...)`) weren't allowed, but should have been. This is now fixed. *[perl #116639]*
- Some regressions from Perl 5.20 have been fixed, in which some syntax errors in *`(?[...])`* constructs within regular expression patterns could cause a segfault instead of a proper error message. *[perl #126180] [perl #126404]*
- Another problem with *`(?[...])`* constructs has been fixed wherein things like `\c]` could cause panics. *[perl #126181]*

- Some problems with attempting to extend the perl stack to around 2G or 4G entries have been fixed. This was particularly an issue on 32-bit perls built to use 64-bit integers, and was easily noticeable with the list repetition operator, e.g.

  ```
      @a = (1) x $big_number
  ```

  Formerly perl may have crashed, depending on the exact value of `$big_number`; now it will typically raise an exception. *[perl #125937]*
- In a regex conditional expression `(?(condition)yes-pattern|no-pattern)`, if the condition is `(?!)` then perl failed the match outright instead of matching the no-pattern. This has been fixed. *[perl #126222]*
- The special backtracking control verbs `(*VERB:ARG)` now all allow an optional argument and set `REGERROR`/`REGMARK` appropriately as well. *[perl #126186]*
- Several bugs, including a segmentation fault, have been fixed with the boundary checking constructs (introduced in Perl 5.22) `\b{gcb}`, `\b{sb}`, `\b{wb}`, `\B{gcb}`, `\B{sb}`, and `\B{wb}`. All the `\B{}` ones now match an empty string; none of the `\b{}` ones do. *[perl #126319]*
- Duplicating a closed file handle for write no longer creates a filename of the form *GLOB(0xXXXXXXXX)*. [perl #125115]
- Warning fatality is now ignored when rewinding the stack. This prevents infinite recursion when the now fatal error also causes rewinding of the stack. [perl #123398]
- In perl v5.22.0, the logic changed when parsing a numeric parameter to the -C option, such that the successfully parsed number was not saved as the option value if it parsed to the end of the argument. [perl #125381]
- The PadlistNAMES macro is an lvalue again.
- Zero -DPERL_TRACE_OPS memory for sub-threads.

  `perl_clone_using()` was missing Zero init of PL_op_exec_cnt[]. This caused sub-threads in threaded -DPERL_TRACE_OPS builds to spew exceedingly large op-counts at destruct. These counts would print %x as "ABABABAB", clearly a mem-poison value.
- A leak in the XS typemap caused one scalar to be leaked each time a `FILE *` or a `PerlIO *` was `OUTPUT:`ed or imported to Perl, since perl 5.000. These particular typemap entries are thought to be extremely rarely used by XS modules. [perl #124181]
- `alarm()` and `sleep()` will now warn if the argument is a negative number and return undef. Previously they would pass the negative value to the underlying C function which may have set up a timer with a surprising value.
- Perl can again be compiled with any Unicode version. This used to (mostly) work, but was lost in v5.18 through v5.20. The property `Name_Alias` did not exist prior to Unicode 5.0. *Unicode::UCD* incorrectly said it did. This has been fixed.
- Very large code-points (beyond Unicode) in regular expressions no longer cause a buffer overflow in some cases when converted to UTF-8. *[perl #125826]*
- The integer overflow check for the range operator (...) in list context now correctly handles the case where the size of the range is larger than the address space. This could happen on 32-bits with -Duse64bitint. *[perl #125781]*
- A crash with `%::=(); J->${\"::"}` has been fixed. *[perl #125541]*
- `qr/(?[ () ])/` no longer segfaults, giving a syntax error message instead. [perl #125805]
- Regular expression possessive quantifier v5.20 regression now fixed. `qr/`*PAT*`{`*min*,*max*`}+/`

is supposed to behave identically to `qr/(?>`*PAT*`{`*min*,*max*`})/`. Since v5.20, this didn't work if *min* and *max* were equal. [perl #125825]

- `BEGIN <>` no longer segfaults and properly produces an error message. [perl #125341]
- In `tr///` an illegal backwards range like `tr/\x{101}-\x{100}//` was not always detected, giving incorrect results. This is now fixed.

Known Problems

- Some modules have been broken by the *context stack rework*. These modules were relying on non-guaranteed implementation details in perl. Their maintainers have been informed, and should contact perl5-porters for advice if needed. Below is a subset of these modules:

  Algorithm::Permute

  Coro

  Coro and perl v5.22.0 were already incompatible due to a change in the perl, and the reworking on the perl context stack creates a further incompatibility. perl5-porters has *discussed the issue on the mailing list*.

  Data::Alias

  RPerl

  Scope::Upper

  TryCatch

- The module *lexical::underscore* no longer works on perl v5.24.0, because perl no longer has a lexical $_!
- `mod_perl` has been patched for compatibility for v5.22.0 and later but no release has been made. The relevant patch (and other changes) can be found in their source code repository, *mirrored at GitHub*.

Acknowledgements

Perl 5.24.0 represents approximately 11 months of development since Perl 5.22.0 and contains approximately 360,000 lines of changes across 1,800 files from 77 authors.

Excluding auto-generated files, documentation and release tools, there were approximately 250,000 lines of changes to 1,200 .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 5.24.0:

Aaron Crane, Aaron Priven, Abigail, Achim Gratz, Alexander D'Archangel, Alex Vandiver, Andreas KÃ¶nig, Andy Broad, Andy Dougherty, Aristotle Pagaltzis, Chase Whitener, Chas. Owens, Chris 'BinGOs' Williams, Craig A. Berry, Dagfinn Ilmari MannsÃ¥ker, Dan Collins, Daniel Dragan, David Golden, David Mitchell, Dominic Hargreaves, Doug Bell, Dr.Ruud, Ed Avis, Ed J, Father Chrysostomos, Herbert Breunung, H.Merijn Brand, Hugo van der Sanden, Ivan Pozdeev, James E Keenan, Jan Dubois, Jarkko Hietaniemi, Jerry D. Hedden, Jim Cromie, John Peacock, John SJ Anderson, Karen Etheridge, Karl Williamson, kmx, Leon Timmermans, Ludovic E. R. Tolhurst-Cleaver, Lukas Mai, Martijn Lievaart, Matthew Horsfall, Mattia Barbon, Max Maischein, Mohammed El-Afifi, Nicholas Clark, Nicolas R., Niko Tyni, Peter John Acklam, Peter Martini, Peter Rabbitson, Pip Cet, Rafael Garcia-Suarez, Reini Urban, Renee Baecker, Ricardo Signes, Sawyer X, Shlomi Fish, Sisyphus, Stanislaw Pusep, Steffen MÃ¼ller, Stevan Little, Steve Hay, Sullivan Beck, Thomas Sibley, Todd Rinaldo, Tom Hukins, Tony Cook, Unicode Consortium, Victor Adam, Vincent Pit, Vladimir Timofeev, Yves Orton, Zachary Storer, Zefram.

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 see *"SECURITY VULNERABILITY CONTACT INFORMATION" in perlsec* for details of how to report the issue.

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.