## NAME

perl5120delta - what is new for perl v5.12.0

## DESCRIPTION

This document describes differences between the 5.10.0 release and the 5.12.0 release.

Many of the bug fixes in 5.12.0 are already included in the 5.10.1 maintenance release.

You can see the list of those changes in the 5.10.1 release notes (*perl5101delta*).

## Core Enhancements

### New package NAME VERSION syntax

This new syntax allows a module author to set the $VERSION of a namespace when the namespace is declared with 'package'. It eliminates the need for `our $VERSION = ...` and similar constructs. E.g.

```
package Foo::Bar 1.23;
# $Foo::Bar::VERSION == 1.23
```

There are several advantages to this:

- `$VERSION` is parsed in exactly the same way as `use NAME VERSION`
- `$VERSION` is set at compile time
- `$VERSION` is a version object that provides proper overloading of comparison operators so comparing `$VERSION` to decimal (1.23) or dotted-decimal (v1.2.3) version numbers works correctly.
- Eliminates `$VERSION = ...` and `eval $VERSION` clutter
- As it requires VERSION to be a numeric literal or v-string literal, it can be statically parsed by toolchain modules without `eval` the way MM->parse_version does for `$VERSION = ...`

It does not break old code with only `package NAME`, but code that uses `package NAME VERSION` will need to be restricted to perl 5.12.0 or newer This is analogous to the change to `open` from two-args to three-args. Users requiring the latest Perl will benefit, and perhaps after several years, it will become a standard practice.

However, `package NAME VERSION` requires a new, 'strict' version number format. See *Version number formats* for details.

### The ... operator

A new operator, `...`, nicknamed the Yada Yada operator, has been added. It is intended to mark placeholder code that is not yet implemented. See *"Yada Yada Operator" in perlop*.

### Implicit strictures

Using the `use VERSION` syntax with a version number greater or equal to 5.11.0 will lexically enable strictures just like `use strict` would do (in addition to enabling features.) The following:

```
use 5.12.0;
```

means:

```
use strict;
use feature ':5.12';
```

### Unicode improvements

Perl 5.12 comes with Unicode 5.2, the latest version available to us at the time of release. This version of Unicode was released in October 2009. See *http://www.unicode.org/versions/Unicode5.2.0* for further details about what's changed in this version of the standard. See *perlunicode* for instructions on installing and using other versions of Unicode.

Additionally, Perl's developers have significantly improved Perl's Unicode implementation. For full details, see *Unicode overhaul* below.

### Y2038 compliance

Perl's core time-related functions are now Y2038 compliant. (It may not mean much to you, but your kids will love it!)

### qr overloading

It is now possible to overload the `qr//` operator, that is, conversion to regexp, like it was already possible to overload conversion to boolean, string or number of objects. It is invoked when an object appears on the right hand side of the =~ operator or when it is interpolated into a regexp. See *overload*.

### Pluggable keywords

Extension modules can now cleanly hook into the Perl parser to define new kinds of keyword-headed expression and compound statement. The syntax following the keyword is defined entirely by the extension. This allows a completely non-Perl sublanguage to be parsed inline, with the correct ops cleanly generated.

See *"PL_keyword_plugin" in perlapi* for the mechanism. The Perl core source distribution also includes a new module *XS::APItest::KeywordRPN*, which implements reverse Polish notation arithmetic via pluggable keywords. This module is mainly used for test purposes, and is not normally installed, but also serves as an example of how to use the new mechanism.

Perl's developers consider this feature to be experimental. We may remove it or change it in a backwards-incompatible way in Perl 5.14.

### APIs for more internals

The lowest layers of the lexer and parts of the pad system now have C APIs available to XS extensions. These are necessary to support proper use of pluggable keywords, but have other uses too. The new APIs are experimental, and only cover a small proportion of what would be necessary to take full advantage of the core's facilities in these areas. It is intended that the Perl 5.13 development cycle will see the addition of a full range of clean, supported interfaces.

Perl's developers consider this feature to be experimental. We may remove it or change it in a backwards-incompatible way in Perl 5.14.

### Overridable function lookup

Where an extension module hooks the creation of rv2cv ops to modify the subroutine lookup process, this now works correctly for bareword subroutine calls. This means that prototypes on subroutines referenced this way will be processed correctly. (Previously bareword subroutine names were initially looked up, for parsing purposes, by an unhookable mechanism, so extensions could only properly influence subroutine names that appeared with an & sigil.)

### A proper interface for pluggable Method Resolution Orders

As of Perl 5.12.0 there is a new interface for plugging and using method resolution orders other than the default linear depth first search. The C3 method resolution order added in 5.10.0 has been re-implemented as a plugin, without changing its Perl-space interface. See *perlmroapi* for more information.

### \N experimental regex escape

Perl now supports `\N`, a new regex escape which you can think of as the inverse of `\n`. It will match any character that is not a newline, independently from the presence or absence of the single line match modifier `/s`. It is not usable within a character class. `\N{3}` means to match 3 non-newlines; `\N{5,}` means to match at least 5. `\N{NAME}` still means the character or sequence named `NAME`, but `NAME` no longer can be things like `3`, or `5,`.

This will break a *custom charnames translator* which allows numbers for character names, as `\N{3}` will now mean to match 3 non-newline characters, and not the character whose name is 3. (No name defined by the Unicode standard is a number, so only custom translators might be affected.)

Perl's developers are somewhat concerned about possible user confusion with the existing `\N{...}` construct which matches characters by their Unicode name. Consequently, this feature is experimental. We may remove it or change it in a backwards-incompatible way in Perl 5.14.

### DTrace support

Perl now has some support for DTrace. See "DTrace support" in *INSTALL*.

### Support for configure_requires in CPAN module metadata

Both `CPAN` and `CPANPLUS` now support the `configure_requires` keyword in the *META.yml* metadata file included in most recent CPAN distributions. This allows distribution authors to specify configuration prerequisites that must be installed before running *Makefile.PL* or *Build.PL*.

See the documentation for `ExtUtils::MakeMaker` or `Module::Build` for more on how to specify `configure_requires` when creating a distribution for CPAN.

### each, keys, values are now more flexible

The `each`, `keys`, `values` function can now operate on arrays.

### when as a statement modifier

`when` is now allowed to be used as a statement modifier.

### $, flexibility

The variable `$,` may now be tied.

### // in when clauses

// now behaves like || in when clauses

### Enabling warnings from your shell environment

You can now set `-W` from the `PERL5OPT` environment variable

### delete local

`delete local` now allows you to locally delete a hash entry.

### New support for Abstract namespace sockets

Abstract namespace sockets are Linux-specific socket type that live in AF_UNIX family, slightly abusing it to be able to use arbitrary character arrays as addresses: They start with nul byte and are not terminated by nul byte, but with the length passed to the socket() system call.

### 32-bit limit on substr arguments removed

The 32-bit limit on `substr` arguments has now been removed. The full range of the system's signed and unsigned integers is now available for the `pos` and `len` arguments.

## Potentially Incompatible Changes

### Deprecations warn by default

Over the years, Perl's developers have deprecated a number of language features for a variety of reasons. Perl now defaults to issuing a warning if a deprecated language feature is used. Many of the deprecations Perl now warns you about have been deprecated for many years. You can find a list of what was deprecated in a given release of Perl in the `perl5xxdelta.pod` file for that release.

To disable this feature in a given lexical scope, you should use `no warnings 'deprecated';` For information about which language features are deprecated and explanations of various deprecation warnings, please see *perldiag*. See *Deprecations* below for the list of features and modules Perl's developers have deprecated as part of this release.

### Version number formats

Acceptable version number formats have been formalized into "strict" and "lax" rules. `package NAME VERSION` takes a strict version number. `UNIVERSAL::VERSION` and the *version* object constructors take lax version numbers. Providing an invalid version will result in a fatal error. The version argument in `use NAME VERSION` is first parsed as a numeric literal or v-string and then passed to `UNIVERSAL::VERSION` (and must then pass the "lax" format test).

These formats are documented fully in the *version* module. To a first approximation, a "strict" version number is a positive decimal number (integer or decimal-fraction) without exponentiation or else a dotted-decimal v-string with a leading 'v' character and at least three components. A "lax" version number allows v-strings with fewer than three components or without a leading 'v'. Under "lax" rules, both decimal and dotted-decimal versions may have a trailing "alpha" component separated by an underscore character after a fractional or dotted-decimal component.

The *version* module adds `version::is_strict` and `version::is_lax` functions to check a scalar against these rules.

### @INC reorganization

In `@INC`, `ARCHLIB` and `PRIVLIB` now occur after the current version's `site_perl` and `vendor_perl`. Modules installed into `site_perl` and `vendor_perl` will now be loaded in preference to those installed in `ARCHLIB` and `PRIVLIB`.

### REGEXPs are now first class

Internally, Perl now treats compiled regular expressions (such as those created with `qr//`) as first class entities. Perl modules which serialize, deserialize or otherwise have deep interaction with Perl's internal data structures need to be updated for this change. Most affected CPAN modules have already been updated as of this writing.

### Switch statement changes

The `given`/`when` switch statement handles complex statements better than Perl 5.10.0 did (These enhancements are also available in 5.10.1 and subsequent 5.10 releases.) There are two new cases where `when` now interprets its argument as a boolean, instead of an expression to be used in a smart match:

flip-flop operators

The `..` and `...` flip-flop operators are now evaluated in boolean context, following their usual semantics; see *"Range Operators" in perlop*.

Note that, as in perl 5.10.0, `when (1..10)` will not work to test whether a given value is an integer between 1 and 10; you should use `when ([1..10])` instead (note the array reference).

However, contrary to 5.10.0, evaluating the flip-flop operators in boolean context ensures it can now be useful in a `when()`, notably for implementing bistable conditions, like in:

```
when (/^=begin/ .. /^=end/) {
  # do something
}
```

defined-or operator

A compound expression involving the defined-or operator, as in `when (expr1 // expr2)`, will be treated as boolean if the first expression is boolean. (This just extends the existing rule that applies to the regular or operator, as in `when (expr1 || expr2)`.)

## Smart match changes

Since Perl 5.10.0, Perl's developers have made a number of changes to the smart match operator. These, of course, also alter the behaviour of the switch statements where smart matching is implicitly used. These changes were also made for the 5.10.1 release, and will remain in subsequent 5.10 releases.

### Changes to type-based dispatch

The smart match operator ~~ is no longer commutative. The behaviour of a smart match now depends primarily on the type of its right hand argument. Moreover, its semantics have been adjusted for greater consistency or usefulness in several cases. While the general backwards compatibility is maintained, several changes must be noted:

- Code references with an empty prototype are no longer treated specially. They are passed an argument like the other code references (even if they choose to ignore it).
- `%hash ~~ sub {}` and `@array ~~ sub {}` now test that the subroutine returns a true value for each key of the hash (or element of the array), instead of passing the whole hash or array as a reference to the subroutine.
- Due to the commutativity breakage, code references are no longer treated specially when appearing on the left of the ~~ operator, but like any vulgar scalar.
- `undef ~~ %hash` is always false (since `undef` can't be a key in a hash). No implicit conversion to `""` is done (as was the case in perl 5.10.0).
- `$scalar ~~ @array` now always distributes the smart match across the elements of the array. It's true if one element in @array verifies `$scalar ~~ $element`. This is a generalization of the old behaviour that tested whether the array contained the scalar.

The full dispatch table for the smart match operator is given in *"Smart matching in detail" in perlsyn*.

### Smart match and overloading

According to the rule of dispatch based on the rightmost argument type, when an object overloading ~~ appears on the right side of the operator, the overload routine will always be called (with a 3rd argument set to a true value, see *overload*.) However, when the object will appear on the left, the overload routine will be called only when the rightmost argument is a simple scalar. This way, distributivity of smart match across arrays is not broken, as well as the other behaviours with complex types (coderefs, hashes, regexes). Thus, writers of overloading routines for smart match mostly need to worry only with comparing against a scalar, and possibly with stringification overloading; the other common cases will be automatically handled consistently.

~~ will now refuse to work on objects that do not overload it (in order to avoid relying on the object's underlying structure). (However, if the object overloads the stringification or the numification operators, and if overload fallback is active, it will be used instead, as usual.)

## Other potentially incompatible changes

- The definitions of a number of Unicode properties have changed to match those of the current Unicode standard. These are listed above under *Unicode overhaul*. This change may break code that expects the old definitions.
- The boolkeys op has moved to the group of hash ops. This breaks binary compatibility.
- Filehandles are now always blessed into `IO::File`.

The previous behaviour was to bless Filehandles into *FileHandle* (an empty proxy class) if it was loaded into memory and otherwise to bless them into `IO::Handle`.

- The semantics of `use feature :5.10*` have changed slightly. See *Modules and Pragmata* for more information.
- Perl's developers now use git, rather than Perforce. This should be a purely internal change only relevant to people actively working on the core. However, you may see minor difference in perl as a consequence of the change. For example in some of details of the output of `perl -V`. See *perlrepository* for more information.
- As part of the `Test::Harness` 2.x to 3.x upgrade, the experimental `Test::Harness::Straps` module has been removed. See *Modules and Pragmata* for more details.
- As part of the `ExtUtils::MakeMaker` upgrade, the `ExtUtils::MakeMaker::bytes` and `ExtUtils::MakeMaker::vmsish` modules have been removed from this distribution.
- `Module::CoreList` no longer contains the `%:patchlevel` hash.
- `length undef` now returns undef.
- Unsupported private C API functions are now declared "static" to prevent leakage to Perl's public API.
- To support the bootstrapping process, *miniperl* no longer builds with UTF-8 support in the regexp engine.

  This allows a build to complete with PERL_UNICODE set and a UTF-8 locale. Without this there's a bootstrapping problem, as miniperl can't load the UTF-8 components of the regexp engine, because they're not yet built.
- *miniperl*'s @INC is now restricted to just `-I...`, the split of `$ENV{PERL5LIB}`, and `"."`
- A space or a newline is now required after a `"#line XXX"` directive.
- Tied filehandles now have an additional method EOF which provides the EOF type.
- To better match all other flow control statements, `foreach` may no longer be used as an attribute.
- Perl's command-line switch "-P", which was deprecated in version 5.10.0, has now been removed. The CPAN module `Filter::cpp` can be used as an alternative.

## Deprecations

From time to time, Perl's developers find it necessary to deprecate features or modules we've previously shipped as part of the core distribution. We are well aware of the pain and frustration that a backwards-incompatible change to Perl can cause for developers building or maintaining software in Perl. You can be sure that when we deprecate a functionality or syntax, it isn't a choice we make lightly. Sometimes, we choose to deprecate functionality or syntax because it was found to be poorly designed or implemented. Sometimes, this is because they're holding back other features or causing performance problems. Sometimes, the reasons are more complex. Wherever possible, we try to keep deprecated functionality available to developers in its previous form for at least one major release. So long as a deprecated feature isn't actively disrupting our ability to maintain and extend Perl, we'll try to leave it in place as long as possible.

The following items are now deprecated:

suidperl

`suidperl` is no longer part of Perl. It used to provide a mechanism to emulate setuid permission bits on systems that don't support it properly.

Use of := to mean an empty attribute list

An accident of Perl's parser meant that these constructions were all equivalent:

```
my $pi := 4;
my $pi : = 4;
my $pi :  = 4;
```

with the : being treated as the start of an attribute list, which ends before the =. As whitespace is not significant here, all are parsed as an empty attribute list, hence all the above are equivalent to, and better written as

```
my $pi = 4;
```

because no attribute processing is done for an empty list.

As is, this meant that := cannot be used as a new token, without silently changing the meaning of existing code. Hence that particular form is now deprecated, and will become a syntax error. If it is absolutely necessary to have empty attribute lists (for example, because of a code generator) then avoid the warning by adding a space before the =.

`UNIVERSAL->import()`

The method `UNIVERSAL->import()` is now deprecated. Attempting to pass import arguments to a `use UNIVERSAL` statement will result in a deprecation warning.

Use of "goto" to jump into a construct

Using `goto` to jump from an outer scope into an inner scope is now deprecated. This rare use case was causing problems in the implementation of scopes.

Custom character names in \N{name} that don't look like names

In `\N{`*name*`}`, *name* can be just about anything. The standard Unicode names have a very limited domain, but a custom name translator could create names that are, for example, made up entirely of punctuation symbols. It is now deprecated to make names that don't begin with an alphabetic character, and aren't alphanumeric or contain other than a very few other characters, namely spaces, dashes, parentheses and colons. Because of the added meaning of `\N` (See *`\N` experimental regex escape*), names that look like curly brace -enclosed quantifiers won't work. For example, `\N{3,4}` now means to match 3 to 4 non-newlines; before a custom name `3,4` could have been created.

Deprecated Modules

The following modules will be removed from the core distribution in a future release, and should be installed from CPAN instead. Distributions on CPAN which require these should add them to their prerequisites. The core versions of these modules warnings will issue a deprecation warning.

If you ship a packaged version of Perl, either alone or as part of a larger system, then you should carefully consider the repercussions of core module deprecations. You may want to consider shipping your default build of Perl with packages for some or all deprecated modules which install into `vendor` or `site` perl library directories. This will inhibit the deprecation warnings.

Alternatively, you may want to consider patching *lib/deprecate.pm* to provide deprecation warnings specific to your packaging system or distribution of Perl, consistent with how your packaging system or distribution manages a staged transition from a release where the installation of a single package provides the given functionality, to a later release where the system administrator needs to know to install multiple packages to get that same functionality.

You can silence these deprecation warnings by installing the modules in question from CPAN. To install the latest version of all of them, just install `Task::Deprecations::5_12`.

*Class::ISA*

*Pod::Plainer*

*Shell*

*Switch*

Switch is buggy and should be avoided. You may find Perl's new `given`/`when` feature a suitable replacement. See *"Switch statements" in perlsyn* for more information.

Assignment to $[

Use of the attribute :locked on subroutines

Use of "locked" with the attributes pragma

Use of "unique" with the attributes pragma

Perl_pmflag

`Perl_pmflag` is no longer part of Perl's public API. Calling it now generates a deprecation warning, and it will be removed in a future release. Although listed as part of the API, it was never documented, and only ever used in *toke.c*, and prior to 5.10, *regcomp.c*. In core, it has been replaced by a static function.

Numerous Perl 4-era libraries

*termcap.pl*, *tainted.pl*, *stat.pl*, *shellwords.pl*, *pwd.pl*, *open3.pl*, *open2.pl*, *newgetopt.pl*, *look.pl*, *find.pl*, *finddepth.pl*, *importenv.pl*, *hostname.pl*, *getopts.pl*, *getopt.pl*, *getcwd.pl*, *flush.pl*, *fastcwd.pl*, *exceptions.pl*, *ctime.pl*, *complete.pl*, *cacheout.pl*, *bigrat.pl*, *bigint.pl*, *bigfloat.pl*, *assert.pl*, *abbrev.pl*, *dotsh.pl*, and *timelocal.pl* are all now deprecated. Earlier, Perl's developers intended to remove these libraries from Perl's core for the 5.14.0 release.

During final testing before the release of 5.12.0, several developers discovered current production code using these ancient libraries, some inside the Perl core itself. Accordingly, the pumpking granted them a stay of execution. They will begin to warn about their deprecation in the 5.14.0 release and will be removed in the 5.16.0 release.

## Unicode overhaul

Perl's developers have made a concerted effort to update Perl to be in sync with the latest Unicode standard. Changes for this include:

Perl can now handle every Unicode character property. New documentation, *perluniprops*, lists all available non-Unihan character properties. By default, perl does not expose Unihan, deprecated or Unicode-internal properties. See below for more details on these; there is also a section in the pod listing them, and explaining why they are not exposed.

Perl now fully supports the Unicode compound-style of using = and `:` in writing regular expressions: `\p{property=value}` and `\p{property:value}` (both of which mean the same thing).

Perl now fully supports the Unicode loose matching rules for text between the braces in `\p{...}` constructs. In addition, Perl allows underscores between digits of numbers.

Perl now accepts all the Unicode-defined synonyms for properties and property values.

`qr/\X/`, which matches a Unicode logical character, has been expanded to work better with various Asian languages. It now is defined as an *extended grapheme cluster*. (See *http://www.unicode.org/reports/tr29/*). Anything matched previously and that made sense will continue to be accepted. Additionally:

- `\X` will not break apart a `CR LF` sequence.
- `\X` will now match a sequence which includes the `ZWJ` and `ZWNJ` characters.
- `\X` will now always match at least one character, including an initial mark. Marks generally come after a base character, but it is possible in Unicode to have them in isolation, and `\X` will now handle that case, for example at the beginning of a line, or after a `ZWSP`. And this is the

part where \X doesn't match the things that it used to that don't make sense. Formerly, for example, you could have the nonsensical case of an accented LF.

- \X will now match a (Korean) Hangul syllable sequence, and the Thai and Lao exception cases.

Otherwise, this change should be transparent for the non-affected languages.

\p{...} matches using the Canonical_Combining_Class property were completely broken in previous releases of Perl. They should now work correctly.

Before Perl 5.12, the Unicode Decomposition_Type=Compat property and a Perl extension had the same name, which led to neither matching all the correct values (with more than 100 mistakes in one, and several thousand in the other). The Perl extension has now been renamed to be Decomposition_Type=Noncanonical (short: dt=noncanon). It has the same meaning as was previously intended, namely the union of all the non-canonical Decomposition types, with Unicode Compat being just one of those.

\p{Decomposition_Type=Canonical} now includes the Hangul syllables.

\p{Uppercase} and \p{Lowercase} now work as the Unicode standard says they should. This means they each match a few more characters than they used to.

\p{Cntrl} now matches the same characters as \p{Control}. This means it no longer will match Private Use (gc=co), Surrogates (gc=cs), nor Format (gc=cf) code points. The Format code points represent the biggest possible problem. All but 36 of them are either officially deprecated or strongly discouraged from being used. Of those 36, likely the most widely used are the soft hyphen (U+00AD), and BOM, ZWSP, ZWNJ, WJ, and similar characters, plus bidirectional controls.

\p{Alpha} now matches the same characters as \p{Alphabetic}. Before 5.12, Perl's definition included a number of things that aren't really alpha (all marks) while omitting many that were. The definitions of \p{Alnum} and \p{Word} depend on Alpha's definition and have changed accordingly.

\p{Word} no longer incorrectly matches non-word characters such as fractions.

\p{Print} no longer matches the line control characters: Tab, LF, CR, FF, VT, and NEL. This brings it in line with standards and the documentation.

\p{XDigit} now matches the same characters as \p{Hex_Digit}. This means that in addition to the characters it currently matches, [A-Fa-f0-9], it will also match the 22 fullwidth equivalents, for example U+FF10: FULLWIDTH DIGIT ZERO.

The Numeric type property has been extended to include the Unihan characters.

There is a new Perl extension, the 'Present_In', or simply 'In', property. This is an extension of the Unicode Age property, but \p{In=5.0} matches any code point whose usage has been determined *as of* Unicode version 5.0. The \p{Age=5.0} only matches code points added in *precisely* version 5.0.

A number of properties now have the correct values for unassigned code points. The affected properties are Bidi_Class, East_Asian_Width, Joining_Type, Decomposition_Type, Hangul_Syllable_Type, Numeric_Type, and Line_Break.

The Default_Ignorable_Code_Point, ID_Continue, and ID_Start properties are now up to date with current Unicode definitions.

Earlier versions of Perl erroneously exposed certain properties that are supposed to be Unicode internal-only. Use of these in regular expressions will now generate, if enabled, a deprecation warning message. The properties are: Other_Alphabetic, Other_Default_Ignorable_Code_Point, Other_Grapheme_Extend, Other_ID_Continue, Other_ID_Start, Other_Lowercase, Other_Math, and

Other_Uppercase.It is now possible to change which Unicode properties Perl understands on a per-installation basis. As mentioned above, certain properties are turned off by default. These include all the Unihan properties (which should be accessible via the CPAN module Unicode::Unihan) and any deprecated or Unicode internal-only property that Perl has never exposed.

The generated files in the `lib/unicore/To` directory are now more clearly marked as being stable, directly usable by applications. New hash entries in them give the format of the normal entries, which allows for easier machine parsing. Perl can generate files in this directory for any property, though most are suppressed. You can find instructions for changing which are written in *perluniprops*.

## Modules and Pragmata

### New Modules and Pragmata

`autodie`

`autodie` is a new lexically-scoped alternative for the `Fatal` module. The bundled version is 2.06_01. Note that in this release, using a string eval when `autodie` is in effect can cause the autodie behaviour to leak into the surrounding scope. See *"BUGS" in autodie* for more details.

Version 2.06_01 has been added to the Perl core.

`Compress::Raw::Bzip2`

Version 2.024 has been added to the Perl core.

`overloading`

`overloading` allows you to lexically disable or enable overloading for some or all operations.

Version 0.001 has been added to the Perl core.

`parent`

`parent` establishes an ISA relationship with base classes at compile time. It provides the key feature of `base` without further unwanted behaviors.

Version 0.223 has been added to the Perl core.

`Parse::CPAN::Meta`

Version 1.40 has been added to the Perl core.

`VMS::DCLsym`

Version 1.03 has been added to the Perl core.

`VMS::Stdio`

Version 2.4 has been added to the Perl core.

`XS::APItest::KeywordRPN`

Version 0.003 has been added to the Perl core.

### Updated Pragmata

`base`

Upgraded from version 2.13 to 2.15.

`bignum`

Upgraded from version 0.22 to 0.23.

`charnames`

`charnames` now contains the Unicode *NameAliases.txt* database file. This has the effect of adding some extra `\N` character names that formerly wouldn't have been recognised; for example, `"\N{LATIN CAPITAL LETTER GHA}"`.

Upgraded from version 1.06 to 1.07.

`constant`

Upgraded from version 1.13 to 1.20.

`diagnostics`

`diagnostics` now supports %.0f formatting internally.

`diagnostics` no longer suppresses `Use of uninitialized value in range (or flip)` warnings. [perl #71204]

Upgraded from version 1.17 to 1.19.

`feature`

In `feature`, the meaning of the `:5.10` and `:5.10.X` feature bundles has changed slightly. The last component, if any (i.e. `X`) is simply ignored. This is predicated on the assumption that new features will not, in general, be added to maintenance releases. So `:5.10` and `:5.10.X` have identical effect. This is a change to the behaviour documented for 5.10.0.

`feature` now includes the `unicode_strings` feature:

```
use feature "unicode_strings";
```

This pragma turns on Unicode semantics for the case-changing operations (`uc`, `lc`, `ucfirst`, `lcfirst`) on strings that don't have the internal UTF-8 flag set, but that contain single-byte characters between 128 and 255.

Upgraded from version 1.11 to 1.16.

`less`

`less` now includes the `stash_name` method to allow subclasses of `less` to pick where in %^H to store their stash.

Upgraded from version 0.02 to 0.03.

`lib`

Upgraded from version 0.5565 to 0.62.

`mro`

`mro` is now implemented as an XS extension. The documented interface has not changed. Code relying on the implementation detail that some `mro::` methods happened to be available at all times gets to "keep both pieces".

Upgraded from version 1.00 to 1.02.

`overload`

`overload` now allow overloading of 'qr'.

Upgraded from version 1.06 to 1.10.

`threads`

Upgraded from version 1.67 to 1.75.

`threads::shared`

Upgraded from version 1.14 to 1.32.

`version`

`version` now has support for *Version number formats* as described earlier in this document and in its own documentation.

Upgraded from version 0.74 to 0.82.

`warnings`

`warnings` has a new `warnings::fatal_enabled()` function. It also includes a new

`illegalproto` warning category. See also *New or Changed Diagnostics* for this change.

Upgraded from version 1.06 to 1.09.

### Updated Modules

`Archive::Extract`

Upgraded from version 0.24 to 0.38.

`Archive::Tar`

Upgraded from version 1.38 to 1.54.

`Attribute::Handlers`

Upgraded from version 0.79 to 0.87.

`AutoLoader`

Upgraded from version 5.63 to 5.70.

`B::Concise`

Upgraded from version 0.74 to 0.78.

`B::Debug`

Upgraded from version 1.05 to 1.12.

`B::Deparse`

Upgraded from version 0.83 to 0.96.

`B::Lint`

Upgraded from version 1.09 to 1.11_01.

`CGI`

Upgraded from version 3.29 to 3.48.

`Class::ISA`

Upgraded from version 0.33 to 0.36.

NOTE: `Class::ISA` is deprecated and may be removed from a future version of Perl.

`Compress::Raw::Zlib`

Upgraded from version 2.008 to 2.024.

`CPAN`

Upgraded from version 1.9205 to 1.94_56.

`CPANPLUS`

Upgraded from version 0.84 to 0.90.

`CPANPLUS::Dist::Build`

Upgraded from version 0.06_02 to 0.46.

`Data::Dumper`

Upgraded from version 2.121_14 to 2.125.

`DB_File`

Upgraded from version 1.816_1 to 1.820.

`Devel::PPPort`

Upgraded from version 3.13 to 3.19.

`Digest`

Upgraded from version 1.15 to 1.16.

`Digest::MD5`

Upgraded from version 2.36_01 to 2.39.

`Digest::SHA`

Upgraded from version 5.45 to 5.47.

`Encode`

Upgraded from version 2.23 to 2.39.

`Exporter`

Upgraded from version 5.62 to 5.64_01.

`ExtUtils::CBuilder`

Upgraded from version 0.21 to 0.27.

`ExtUtils::Command`

Upgraded from version 1.13 to 1.16.

`ExtUtils::Constant`

Upgraded from version 0.2 to 0.22.

`ExtUtils::Install`

Upgraded from version 1.44 to 1.55.

`ExtUtils::MakeMaker`

Upgraded from version 6.42 to 6.56.

`ExtUtils::Manifest`

Upgraded from version 1.51_01 to 1.57.

`ExtUtils::ParseXS`

Upgraded from version 2.18_02 to 2.21.

`File::Fetch`

Upgraded from version 0.14 to 0.24.

`File::Path`

Upgraded from version 2.04 to 2.08_01.

`File::Temp`

Upgraded from version 0.18 to 0.22.

`Filter::Simple`

Upgraded from version 0.82 to 0.84.

`Filter::Util::Call`

Upgraded from version 1.07 to 1.08.

`Getopt::Long`

Upgraded from version 2.37 to 2.38.

`IO`

Upgraded from version 1.23_01 to 1.25_02.

`IO::Zlib`

Upgraded from version 1.07 to 1.10.

`IPC::Cmd`

Upgraded from version 0.40_1 to 0.54.

`IPC::SysV`

Upgraded from version 1.05 to 2.01.

`Locale::Maketext`

Upgraded from version 1.12 to 1.14.

`Locale::Maketext::Simple`

Upgraded from version 0.18 to 0.21.

`Log::Message`

Upgraded from version 0.01 to 0.02.

`Log::Message::Simple`

Upgraded from version 0.04 to 0.06.

`Math::BigInt`

Upgraded from version 1.88 to 1.89_01.

`Math::BigInt::FastCalc`

Upgraded from version 0.16 to 0.19.

`Math::BigRat`

Upgraded from version 0.21 to 0.24.

`Math::Complex`

Upgraded from version 1.37 to 1.56.

`Memoize`

Upgraded from version 1.01_02 to 1.01_03.

`MIME::Base64`

Upgraded from version 3.07_01 to 3.08.

`Module::Build`

Upgraded from version 0.2808_01 to 0.3603.

`Module::CoreList`

Upgraded from version 2.12 to 2.29.

`Module::Load`

Upgraded from version 0.12 to 0.16.

`Module::Load::Conditional`

Upgraded from version 0.22 to 0.34.

`Module::Loaded`

Upgraded from version 0.01 to 0.06.

`Module::Pluggable`

Upgraded from version 3.6 to 3.9.

`Net::Ping`

Upgraded from version 2.33 to 2.36.

`NEXT`

Upgraded from version 0.60_01 to 0.64.

`Object::Accessor`

Upgraded from version 0.32 to 0.36.

`Package::Constants`

Upgraded from version 0.01 to 0.02.

`PerlIO`

Upgraded from version 1.04 to 1.06.

`Pod::Parser`

Upgraded from version 1.35 to 1.37.

`Pod::Perldoc`

Upgraded from version 3.14_02 to 3.15_02.

`Pod::Plainer`

Upgraded from version 0.01 to 1.02.

NOTE: `Pod::Plainer` is deprecated and may be removed from a future version of Perl.

`Pod::Simple`

Upgraded from version 3.05 to 3.13.

`Safe`

Upgraded from version 2.12 to 2.22.

`SelfLoader`

Upgraded from version 1.11 to 1.17.

`Storable`

Upgraded from version 2.18 to 2.22.

`Switch`

Upgraded from version 2.13 to 2.16.

NOTE: `Switch` is deprecated and may be removed from a future version of Perl.

`Sys::Syslog`

Upgraded from version 0.22 to 0.27.

`Term::ANSIColor`

Upgraded from version 1.12 to 2.02.

`Term::UI`

Upgraded from version 0.18 to 0.20.

`Test`

Upgraded from version 1.25 to 1.25_02.

`Test::Harness`

Upgraded from version 2.64 to 3.17.

`Test::Simple`
: Upgraded from version 0.72 to 0.94.

`Text::Balanced`
: Upgraded from version 2.0.0 to 2.02.

`Text::ParseWords`
: Upgraded from version 3.26 to 3.27.

`Text::Soundex`
: Upgraded from version 3.03 to 3.03_01.

`Thread::Queue`
: Upgraded from version 2.00 to 2.11.

`Thread::Semaphore`
: Upgraded from version 2.01 to 2.09.

`Tie::RefHash`
: Upgraded from version 1.37 to 1.38.

`Time::HiRes`
: Upgraded from version 1.9711 to 1.9719.

`Time::Local`
: Upgraded from version 1.18 to 1.1901_01.

`Time::Piece`
: Upgraded from version 1.12 to 1.15.

`Unicode::Collate`
: Upgraded from version 0.52 to 0.52_01.

`Unicode::Normalize`
: Upgraded from version 1.02 to 1.03.

`Win32`
: Upgraded from version 0.34 to 0.39.

`Win32API::File`
: Upgraded from version 0.1001_01 to 0.1101.

`XSLoader`
: Upgraded from version 0.08 to 0.10.

### Removed Modules and Pragmata

`attrs`
: Removed from the Perl core. Prior version was 1.02.

`CPAN::API::HOWTO`
: Removed from the Perl core. Prior version was 'undef'.

`CPAN::DeferedCode`
: Removed from the Perl core. Prior version was 5.50.

`CPANPLUS::inc`

Removed from the Perl core. Prior version was 'undef'.

`DCLsym`

Removed from the Perl core. Prior version was 1.03.

`ExtUtils::MakeMaker::bytes`

Removed from the Perl core. Prior version was 6.42.

`ExtUtils::MakeMaker::vmsish`

Removed from the Perl core. Prior version was 6.42.

`Stdio`

Removed from the Perl core. Prior version was 2.3.

`Test::Harness::Assert`

Removed from the Perl core. Prior version was 0.02.

`Test::Harness::Iterator`

Removed from the Perl core. Prior version was 0.02.

`Test::Harness::Point`

Removed from the Perl core. Prior version was 0.01.

`Test::Harness::Results`

Removed from the Perl core. Prior version was 0.01.

`Test::Harness::Straps`

Removed from the Perl core. Prior version was 0.26_01.

`Test::Harness::Util`

Removed from the Perl core. Prior version was 0.01.

`XSSymSet`

Removed from the Perl core. Prior version was 1.1.

### Deprecated Modules and Pragmata

See *Deprecated Modules* above.

## Documentation

### New Documentation

- *perlhaiku* contains instructions on how to build perl for the Haiku platform.
- *perlmroapi* describes the new interface for pluggable Method Resolution Orders.
- *perlperf*, by Richard Foley, provides an introduction to the use of performance and optimization techniques which can be used with particular reference to perl programs.
- *perlrepository* describes how to access the perl source using the *git* version control system.
- *perlpolicy* extends the "Social contract about contributed modules" into the beginnings of a document on Perl porting policies.

### Changes to Existing Documentation

- The various large *Changes** files (which listed every change made to perl over the last 18 years) have been removed, and replaced by a small file, also called *Changes*, which just explains how that same information may be extracted from the git version control system.

- *Porting/patching.pod* has been deleted, as it mainly described interacting with the old Perforce-based repository, which is now obsolete. Information still relevant has been moved to *perlrepository*.
- The syntax `unless (EXPR) BLOCK else BLOCK` is now documented as valid, as is the syntax `unless (EXPR) BLOCK elsif (EXPR) BLOCK ... else BLOCK`, although actually using the latter may not be the best idea for the readability of your source code.
- Documented -X overloading.
- Documented that `when()` treats specially most of the filetest operators
- Documented `when` as a syntax modifier.
- Eliminated "Old Perl threads tutorial", which described 5005 threads.

  *pod/perlthrtut.pod* is the same material reworked for ithreads.
- Correct previous documentation: v-strings are not deprecated

  With version objects, we need them to use MODULE VERSION syntax. This patch removes the deprecation notice.
- Security contact information is now part of *perlsec*.
- A significant fraction of the core documentation has been updated to clarify the behavior of Perl's Unicode handling.

  Much of the remaining core documentation has been reviewed and edited for clarity, consistent use of language, and to fix the spelling of Tom Christiansen's name.
- The Pod specification (*perlpodspec*) has been updated to bring the specification in line with modern usage already supported by most Pod systems. A parameter string may now follow the format name in a "begin/end" region. Links to URIs with a text description are now allowed. The usage of `L<"section">` has been marked as deprecated.
- *if.pm* has been documented in *"use" in perlfunc* as a means to get conditional loading of modules despite the implicit BEGIN block around `use`.
- The documentation for `$1` in perlvar.pod has been clarified.
- `\N{U+`*`code point`*`}` is now documented.

## Selected Performance Enhancements

- A new internal cache means that `isa()` will often be faster.
- The implementation of C3 Method Resolution Order has been optimised - linearisation for classes with single inheritance is 40% faster. Performance for multiple inheritance is unchanged.
- Under `use locale`, the locale-relevant information is now cached on read-only values, such as the list returned by `keys %hash`. This makes operations such as `sort keys %hash` in the scope of `use locale` much faster.
- Empty `DESTROY` methods are no longer called.
- `Perl_sv_utf8_upgrade()` is now faster.
- `keys` on empty hash is now faster.
- `if (%foo)` has been optimized to be faster than `if (keys %foo)`.
- The string repetition operator (`$str x $num`) is now several times faster when `$str` has length one or `$num` is large.

- Reversing an array to itself (as in `@a = reverse @a`) in void context now happens in-place and is several orders of magnitude faster than it used to be. It will also preserve non-existent elements whenever possible, i.e. for non magical arrays or tied arrays with `EXISTS` and `DELETE` methods.

## Installation and Configuration Improvements

- *perlapi*, *perlintern*, *perlmodlib* and *perltoc* are now all generated at build time, rather than being shipped as part of the release.
- If `vendorlib` and `vendorarch` are the same, then they are only added to `@INC` once.
- `$Config{usedevel}` and the C-level `PERL_USE_DEVEL` are now defined if perl is built with `-Dusedevel`.
- *Configure* will enable use of `-fstack-protector`, to provide protection against stack-smashing attacks, if the compiler supports it.
- *Configure* will now determine the correct prototypes for re-entrant functions and for `gconvert` if you are using a C++ compiler rather than a C compiler.
- On Unix, if you build from a tree containing a git repository, the configuration process will note the commit hash you have checked out, for display in the output of `perl -v` and `perl -V`. Unpushed local commits are automatically added to the list of local patches displayed by `perl -V`.
- Perl now supports SystemTap's `dtrace` compatibility layer and an issue with linking `miniperl` has been fixed in the process.
- perldoc now uses `less -R` instead of `less` for improved behaviour in the face of `groff`'s new usage of ANSI escape codes.
- `perl -V` now reports use of the compile-time options `USE_PERL_ATOF` and `USE_ATTRIBUTES_FOR_PERLIO`.
- As part of the flattening of *ext*, all extensions on all platforms are built by *make_ext.pl*. This replaces the Unix-specific *ext/util/make_ext*, VMS-specific *make_ext.com* and Win32-specific *win32/buildext.pl*.

## Internal Changes

Each release of Perl sees numerous internal changes which shouldn't affect day to day usage but may still be notable for developers working with Perl's source code.

- The J.R.R. Tolkien quotes at the head of C source file have been checked and proper citations added, thanks to a patch from Tom Christiansen.
- The internal structure of the dual-life modules traditionally found in the *lib/* and *ext/* directories in the perl source has changed significantly. Where possible, dual-lifed modules have been extracted from *lib/* and *ext/*.

  Dual-lifed modules maintained by Perl's developers as part of the Perl core now live in *dist/*. Dual-lifed modules maintained primarily on CPAN now live in *cpan/*. When reporting a bug in a module located under *cpan/*, please send your bug report directly to the module's bug tracker or author, rather than Perl's bug tracker.
- `\N{...}` now compiles better, always forces UTF-8 internal representation

  Perl's developers have fixed several problems with the recognition of `\N{...}` constructs. As part of this, perl will store any scalar or regex containing `\N{`*`name`*`}` or `\N{U+`*`code point`*`}` in its definition in UTF-8 format. (This was true previously for all occurrences of `\N{`*`name`*`}` that did not use a custom translator, but now it's always true.)
- Perl_magic_setmglob now knows about globs, fixing RT #71254.

- SVt_RV no longer exists. RVs are now stored in IVs.
- Perl_vcroak() now accepts a null first argument. In addition, a full audit was made of the "not NULL" compiler annotations, and those for several other internal functions were corrected.
- New macros dSAVEDERRNO, dSAVE_ERRNO, SAVE_ERRNO, RESTORE_ERRNO have been added to formalise the temporary saving of the errno variable.
- The function Perl_sv_insert_flags has been added to augment Perl_sv_insert.
- The function Perl_newSV_type(type) has been added, equivalent to Perl_newSV() followed by Perl_sv_upgrade(type).
- The function Perl_newSVpvn_flags() has been added, equivalent to Perl_newSVpvn() and then performing the action relevant to the flag.

  Two flag bits are currently supported.

  - SVf_UTF8 will call SvUTF8_on() for you. (Note that this does not convert an sequence of ISO 8859-1 characters to UTF-8). A wrapper, newSVpvn_utf8() is available for this.
  - SVs_TEMP now calls Perl_sv_2mortal() on the new SV.

  There is also a wrapper that takes constant strings, newSVpvs_flags().
- The function Perl_croak_xs_usage has been added as a wrapper to Perl_croak.
- Perl now exports the functions PerlIO_find_layer and PerlIO_list_alloc.
- PL_na has been exterminated from the core code, replaced by local STRLEN temporaries, or *_nolen() calls. Either approach is faster than PL_na, which is a pointer dereference into the interpreter structure under ithreads, and a global variable otherwise.
- Perl_mg_free() used to leave freed memory accessible via SvMAGIC() on the scalar. It now updates the linked list to remove each piece of magic as it is freed.
- Under ithreads, the regex in PL_reg_curpm is now reference counted. This eliminates a lot of hackish workarounds to cope with it not being reference counted.
- Perl_mg_magical() would sometimes incorrectly turn on SvRMAGICAL(). This has been fixed.
- The *public* IV and NV flags are now not set if the string value has trailing "garbage". This behaviour is consistent with not setting the public IV or NV flags if the value is out of range for the type.
- Uses of Nullav, Nullcv, Nullhv, Nullop, Nullsv etc have been replaced by NULL in the core code, and non-dual-life modules, as NULL is clearer to those unfamiliar with the core code.
- A macro MUTABLE_PTR(p) has been added, which on (non-pedantic) gcc will not cast away const, returning a void *. Macros MUTABLE_SV(av), MUTABLE_SV(cv) etc build on this, casting to AV * etc without casting away const. This allows proper compile-time auditing of const correctness in the core, and helped picked up some errors (now fixed).
- Macros mPUSHs() and mXPUSHs() have been added, for pushing SVs on the stack and mortalizing them.
- Use of the private structure mro_meta has changed slightly. Nothing outside the core should be accessing this directly anyway.
- A new tool, *Porting/expand-macro.pl* has been added, that allows you to view how a C

preprocessor macro would be expanded when compiled. This is handy when trying to decode the macro hell that is the perl guts.

# Testing

## Testing improvements

Parallel tests

The core distribution can now run its regression tests in parallel on Unix-like platforms. Instead of running `make test`, set `TEST_JOBS` in your environment to the number of tests to run in parallel, and run `make test_harness`. On a Bourne-like shell, this can be done as

```
TEST_JOBS=3 make test_harness  # Run 3 tests in parallel
```

An environment variable is used, rather than parallel make itself, because *TAP::Harness* needs to be able to schedule individual non-conflicting test scripts itself, and there is no standard interface to `make` utilities to interact with their job schedulers.

Note that currently some test scripts may fail when run in parallel (most notably `ext/IO/t/io_dir.t`). If necessary run just the failing scripts again sequentially and see if the failures go away.

Test harness flexibility

It's now possible to override `PERL50PT` and friends in *t/TEST*

Test watchdog

Several tests that have the potential to hang forever if they fail now incorporate a "watchdog" functionality that will kill them after a timeout, which helps ensure that `make test` and `make test_harness` run to completion automatically.

## New Tests

Perl's developers have added a number of new tests to the core. In addition to the items listed below, many modules updated from CPAN incorporate new tests.

- Significant cleanups to core tests to ensure that language and interpreter features are not used before they're tested.
- `make test_porting` now runs a number of important pre-commit checks which might be of use to anyone working on the Perl core.
- *t/porting/podcheck.t* automatically checks the well-formedness of POD found in all .pl, .pm and .pod files in the *MANIFEST*, other than in dual-lifed modules which are primarily maintained outside the Perl core.
- *t/porting/manifest.t* now tests that all files listed in MANIFEST are present.
- *t/op/while_readdir.t* tests that a bare readdir in while loop sets $_.
- *t/comp/retainedlines.t* checks that the debugger can retain source lines from `eval`.
- *t/io/perlio_fail.t* checks that bad layers fail.
- *t/io/perlio_leaks.t* checks that PerlIO layers are not leaking.
- *t/io/perlio_open.t* checks that certain special forms of open work.
- *t/io/perlio.t* includes general PerlIO tests.
- *t/io/pvbm.t* checks that there is no unexpected interaction between the internal types `PVBM` and `PVGV`.
- *t/mro/package_aliases.t* checks that mro works properly in the presence of aliased packages.

- *t/op/dbm.t* tests `dbmopen` and `dbmclose`.
- *t/op/index_thr.t* tests the interaction of `index` and threads.
- *t/op/pat_thr.t* tests the interaction of esoteric patterns and threads.
- *t/op/qr_gc.t* tests that `qr` doesn't leak.
- *t/op/reg_email_thr.t* tests the interaction of regex recursion and threads.
- *t/op/regexp_qr_embed_thr.t* tests the interaction of patterns with embedded `qr//` and threads.
- *t/op/regexp_unicode_prop.t* tests Unicode properties in regular expressions.
- *t/op/regexp_unicode_prop_thr.t* tests the interaction of Unicode properties and threads.
- *t/op/reg_nc_tie.t* tests the tied methods of `Tie::Hash::NamedCapture`.
- *t/op/reg_posixcc.t* checks that POSIX character classes behave consistently.
- *t/op/re.t* checks that exportable `re` functions in *universal.c* work.
- *t/op/setpgrpstack.t* checks that `setpgrp` works.
- *t/op/substr_thr.t* tests the interaction of `substr` and threads.
- *t/op/upgrade.t* checks that upgrading and assigning scalars works.
- *t/uni/lex_utf8.t* checks that Unicode in the lexer works.
- *t/uni/tie.t* checks that Unicode and `tie` work.
- *t/comp/final_line_num.t* tests whether line numbers are correct at EOF
- *t/comp/form_scope.t* tests format scoping.
- *t/comp/line_debug.t* tests whether `@{"_<$file"}` works.
- *t/op/filetest_t.t* tests if -t file test works.
- *t/op/qr.t* tests `qr`.
- *t/op/utf8cache.t* tests malfunctions of the utf8 cache.
- *t/re/uniprops.t* test unicodes `\p{}` regex constructs.
- *t/op/filehandle.t* tests some suitably portable filetest operators to check that they work as expected, particularly in the light of some internal changes made in how filehandles are blessed.
- *t/op/time_loop.t* tests that unix times greater than `2**63`, which can now be handed to `gmtime` and `localtime`, do not cause an internal overflow or an excessively long loop.

## New or Changed Diagnostics

### New Diagnostics

- SV allocation tracing has been added to the diagnostics enabled by `-Dm`. The tracing can alternatively output via the `PERL_MEM_LOG` mechanism, if that was enabled when the *perl* binary was compiled.
- Smartmatch resolution tracing has been added as a new diagnostic. Use `-DM` to enable it.
- A new debugging flag `-DB` now dumps subroutine definitions, leaving `-Dx` for its original purpose of dumping syntax trees.

- Perl 5.12 provides a number of new diagnostic messages to help you write better code. See *perldiag* for details of these new messages.
  - `Bad plugin affecting keyword '%s'`
  - `gmtime(%.0f) too large`
  - `Lexing code attempted to stuff non-Latin-1 character into Latin-1 input`
  - `Lexing code internal error (%s)`
  - `localtime(%.0f) too large`
  - `Overloaded dereference did not return a reference`
  - `Overloaded qr did not return a REGEXP`
  - `Perl_pmflag() is deprecated, and will be removed from the XS API`
  - `lvalue attribute ignored after the subroutine has been defined`

    This new warning is issued when one attempts to mark a subroutine as lvalue after it has been defined.
  - Perl now warns you if ++ or `--` are unable to change the value because it's beyond the limit of representation.

    This uses a new warnings category: "imprecision".
  - `lc`, uc, `lcfirst`, and `ucfirst` warn when passed undef.
  - `Show constant in "Useless use of a constant in void context"`
  - `Prototype after '%s'`
  - `panic: sv_chop %s`

    This new fatal error occurs when the C routine `Perl_sv_chop()` was passed a position that is not within the scalar's string buffer. This could be caused by buggy XS code, and at this point recovery is not possible.
  - The fatal error `Malformed UTF-8 returned by \N` is now produced if the `charnames` handler returns malformed UTF-8.
  - If an unresolved named character or sequence was encountered when compiling a regex pattern then the fatal error `\N{NAME} must be resolved by the lexer` is now produced. This can happen, for example, when using a single-quotish context like `$re = '\N{SPACE}'; /$re/;`. See *perldiag* for more examples of how the lexer can get bypassed.
  - `Invalid hexadecimal number in \N{U+...}` is a new fatal error triggered when the character constant represented by `...` is not a valid hexadecimal number.
  - The new meaning of `\N` as `[^\n]` is not valid in a bracketed character class, just like `.` in a character class loses its special meaning, and will cause the fatal error `\N in a character class must be a named character: \N{...}`.
  - The rules on what is legal for the `...` in `\N{...}` have been tightened up so that unless the `...` begins with an alphabetic character and continues with a combination of alphanumerics, dashes, spaces, parentheses or colons then the warning `Deprecated character(s) in \N{...} starting at '%s'` is now issued.
  - The warning `Using just the first characters returned by \N{}` will be issued if the `charnames` handler returns a sequence of characters which exceeds the

limit of the number of characters that can be used. The message will indicate which characters were used and which were discarded.

## Changed Diagnostics

A number of existing diagnostic messages have been improved or corrected:

- A new warning category `illegalproto` allows finer-grained control of warnings around function prototypes.

  The two warnings:

  ```
  Illegal character in prototype for %s : %s
  Prototype after '%c' for %s : %s
  ```

  have been moved from the `syntax` top-level warnings category into a new first-level category, `illegalproto`. These two warnings are currently the only ones emitted during parsing of an invalid/illegal prototype, so one can now use

  ```
  no warnings 'illegalproto';
  ```

  to suppress only those, but not other syntax-related warnings. Warnings where prototypes are changed, ignored, or not met are still in the `prototype` category as before.

- `Deep recursion on subroutine "%s"`

  It is now possible to change the depth threshold for this warning from the default of 100, by recompiling the *perl* binary, setting the C pre-processor macro `PERL_SUB_DEPTH_WARN` to the desired value.

- `Illegal character in prototype` warning is now more precise when reporting illegal characters after _

- mro merging error messages are now very similar to those produced by *Algorithm::C3*.

- Amelioration of the error message "Unrecognized character %s in column %d"

  Changes the error message to "Unrecognized character %s; marked by <-- HERE after %s<-- HERE near column %d". This should make it a little simpler to spot and correct the suspicious character.

- Perl now explicitly points to `$.` when it causes an uninitialized warning for ranges in scalar context.

- `split` now warns when called in void context.

- `printf`-style functions called with too few arguments will now issue the warning `"Missing argument in %s"` [perl #71000]

- Perl now properly returns a syntax error instead of segfaulting if `each`, `keys`, or `values` is used without an argument.

- `tell()` now fails properly if called without an argument and when no previous file was read.

  `tell()` now returns `-1`, and sets errno to EBADF, thus restoring the 5.8.x behaviour.

- `overload` no longer implicitly unsets fallback on repeated 'use overload' lines.

- POSIX::strftime() can now handle Unicode characters in the format string.

- The `syntax` category was removed from 5 warnings that should only be in `deprecated`.

- Three fatal `pack`/`unpack` error messages have been normalized to `panic: %s`

- `Unicode character is illegal` has been rephrased to be more accurate

It now reads `Unicode non-character is illegal in interchange` and the perldiag documentation has been expanded a bit.

- Currently, all but the first of the several characters that the `charnames` handler may return are discarded when used in a regular expression pattern bracketed character class. If this happens then the warning `Using just the first character returned by \N{} in character class` will be issued.

- The warning `Missing right brace on \N{} or unescaped left brace after \N. Assuming the latter` will be issued if Perl encounters a `\N{` but doesn't find a matching `}`. In this case Perl doesn't know if it was mistakenly omitted, or if "match non-newline" followed by "match a `{`" was desired. It assumes the latter because that is actually a valid interpretation as written, unlike the other case. If you meant the former, you need to add the matching right brace. If you did mean the latter, you can silence this warning by writing instead `\N\{`.

- `gmtime` and `localtime` called with numbers smaller than they can reliably handle will now issue the warnings `gmtime(%.0f) too small` and `localtime(%.0f) too small`.

The following diagnostic messages have been removed:

- `Runaway format`

- `Can't locate package %s for the parents of %s`

  In general this warning it only got produced in conjunction with other warnings, and removing it allowed an ISA lookup optimisation to be added.

- `v-string in use/require is non-portable`

## Utility Changes

- *h2ph* now looks in `include-fixed` too, which is a recent addition to gcc's search path.

- *h2xs* no longer incorrectly treats enum values like macros. It also now handles C++ style comments (`//`) properly in enums.

- *perl5db.pl* now supports LVALUE subroutines. Additionally, the debugger now correctly handles proxy constant subroutines, and subroutine stubs.

- *perlbug* now uses `%Module::CoreList::bug_tracker` to print out upstream bug tracker URLs. If a user identifies a particular module as the topic of their bug report and we're able to divine the URL for its upstream bug tracker, perlbug now provide a message to the user explaining that the core copies the CPAN version directly, and provide the URL for reporting the bug directly to the upstream author.

  *perlbug* no longer reports "Message sent" when it hasn't actually sent the message

- *perlthanks* is a new utility for sending non-bug-reports to the authors and maintainers of Perl. Getting nothing but bug reports can become a bit demoralising. If Perl 5.12 works well for you, please try out *perlthanks*. It will make the developers smile.

- Perl's developers have fixed bugs in *a2p* having to do with the `match()` operator in list context. Additionally, *a2p* no longer generates code that uses the `$[` variable.

## Selected Bug Fixes

- U+0FFFF is now a legal character in regular expressions.

- pp_qr now always returns a new regexp SV. Resolves RT #69852.

  Instead of returning a(nother) reference to the (pre-compiled) regexp in the optree, use reg_temp_copy() to create a copy of it, and return a reference to that. This resolves issues about Regexp::DESTROY not being called in a timely fashion (the original bug tracked by RT

#69852), as well as bugs related to blessing regexps, and of assigning to regexps, as described in correspondence added to the ticket.

It transpires that we also need to undo the SvPVX() sharing when ithreads cloning a Regexp SV, because mother_re is set to NULL, instead of a cloned copy of the mother_re. This change might fix bugs with regexps and threads in certain other situations, but as yet neither tests nor bug reports have indicated any problems, so it might not actually be an edge case that it's possible to reach.

- Several compilation errors and segfaults when perl was built with `-Dmad` were fixed.
- Fixes for lexer API changes in 5.11.2 which broke NYTProf's savesrc option.
- `-t` should only return TRUE for file handles connected to a TTY

  The Microsoft C version of `isatty()` returns TRUE for all character mode devices, including the */dev/null*-style "nul" device and printers like "lpt1".
- Fixed a regression caused by commit fafafbaf which caused a panic during parameter passing [perl #70171]
- On systems which in-place edits without backup files, -i'*' now works as the documentation says it does [perl #70802]
- Saving and restoring magic flags no longer loses readonly flag.
- The malformed syntax `grep EXPR LIST` (note the missing comma) no longer causes abrupt and total failure.
- Regular expressions compiled with `qr{}` literals properly set `$'` when matching again.
- Using named subroutines with `sort` should no longer lead to bus errors [perl #71076]
- Numerous bugfixes catch small issues caused by the recently-added Lexer API.
- Smart match against `@_` sometimes gave false negatives. [perl #71078]
- $@ may now be assigned a read-only value (without error or busting the stack).
- `sort` called recursively from within an active comparison subroutine no longer causes a bus error if run multiple times. [perl #71076]
- Tie::Hash::NamedCapture::* will not abort if passed bad input (RT #71828)
- @_ and $_ no longer leak under threads (RT #34342 and #41138, also #70602, #70974)
- `-I` on shebang line now adds directories in front of @INC as documented, and as does `-I` when specified on the command-line.
- `kill` is now fatal when called on non-numeric process identifiers. Previously, an `undef` process identifier would be interpreted as a request to kill process 0, which would terminate the current process group on POSIX systems. Since process identifiers are always integers, killing a non-numeric process is now fatal.
- 5.10.0 inadvertently disabled an optimisation, which caused a measurable performance drop in list assignment, such as is often used to assign function parameters from `@_`. The optimisation has been re-instated, and the performance regression fixed. (This fix is also present in 5.10.1)
- Fixed memory leak on `while (1) { map 1, 1 }` [RT #53038].
- Some potential coredumps in PerlIO fixed [RT #57322,54828].
- The debugger now works with lvalue subroutines.

- The debugger's `m` command was broken on modules that defined constants [RT #61222].
- `crypt` and string complement could return tainted values for untainted arguments [RT #59998].
- The `-i`*suffix* command-line switch now recreates the file using restricted permissions, before changing its mode to match the original file. This eliminates a potential race condition [RT #60904].
- On some Unix systems, the value in `$?` would not have the top bit set (`$? & 128`) even if the child core dumped.
- Under some circumstances, `$^R` could incorrectly become undefined [RT #57042].
- In the XS API, various hash functions, when passed a pre-computed hash where the key is UTF-8, might result in an incorrect lookup.
- XS code including *XSUB.h* before *perl.h* gave a compile-time error [RT #57176].
- `$object->isa('Foo')` would report false if the package `Foo` didn't exist, even if the object's `@ISA` contained `Foo`.
- Various bugs in the new-to 5.10.0 mro code, triggered by manipulating `@ISA`, have been found and fixed.
- Bitwise operations on references could crash the interpreter, e.g. `$x=\$y; $x |= "foo"` [RT #54956].
- Patterns including alternation might be sensitive to the internal UTF-8 representation, e.g.

```
my $byte = chr(192);
my $utf8 = chr(192); utf8::upgrade($utf8);
$utf8 =~ /$byte|X}/i; # failed in 5.10.0
```

- Within UTF8-encoded Perl source files (i.e. where `use utf8` is in effect), double-quoted literal strings could be corrupted where a `\xNN`, `\0NNN` or `\N{}` is followed by a literal character with ordinal value greater than 255 [RT #59908].
- `B::Deparse` failed to correctly deparse various constructs: `readpipe STRING` [RT #62428], `CORE::require(STRING)` [RT #62488], `sub foo(_)` [RT #62484].
- Using `setpgrp` with no arguments could corrupt the perl stack.
- The block form of `eval` is now specifically trappable by `Safe` and `ops`. Previously it was erroneously treated like string `eval`.
- In 5.10.0, the two characters `[~` were sometimes parsed as the smart match operator (`~~`) [RT #63854].
- In 5.10.0, the `*` quantifier in patterns was sometimes treated as `{0,32767}` [RT #60034, #60464]. For example, this match would fail:

```
("ab" x 32768) =~ /^(ab)*$/
```

- `shmget` was limited to a 32 bit segment size on a 64 bit OS [RT #63924].
- Using `next` or `last` to exit a `given` block no longer produces a spurious warning like the following:

```
Exiting given via last at foo.pl line 123
```

- Assigning a format to a glob could corrupt the format; e.g.:

```
*bar=*foo{FORMAT}; # foo format now bad
```

- Attempting to coerce a typeglob to a string or number could cause an assertion failure. The correct error message is now generated, `Can't coerce GLOB to $type`.
- Under `use filetest 'access'`, `-x` was using the wrong access mode. This has been fixed [RT #49003].
- `length` on a tied scalar that returned a Unicode value would not be correct the first time. This has been fixed.
- Using an array `tie` inside in array `tie` could SEGV. This has been fixed. [RT #51636]
- A race condition inside `PerlIOStdio_close()` has been identified and fixed. This used to cause various threading issues, including SEGVs.
- In `unpack`, the use of `()` groups in scalar context was internally placing a list on the interpreter's stack, which manifested in various ways, including SEGVs. This is now fixed [RT #50256].
- Magic was called twice in `substr`, `\&$x`, `tie $x, $m` and `chop`. These have all been fixed.
- A 5.10.0 optimisation to clear the temporary stack within the implicit loop of `s///ge` has been reverted, as it turned out to be the cause of obscure bugs in seemingly unrelated parts of the interpreter [commit ef0d4e17921ee3de].
- The line numbers for warnings inside `elsif` are now correct.
- The `..` operator now works correctly with ranges whose ends are at or close to the values of the smallest and largest integers.
- `binmode STDIN, ':raw'` could lead to segmentation faults on some platforms. This has been fixed [RT #54828].
- An off-by-one error meant that `index $str, ...` was effectively being executed as `index "$str\0", ...`. This has been fixed [RT #53746].
- Various leaks associated with named captures in regexes have been fixed [RT #57024].
- A weak reference to a hash would leak. This was affecting `DBI` [RT #56908].
- Using (?|) in a regex could cause a segfault [RT #59734].
- Use of a UTF-8 `tr//` within a closure could cause a segfault [RT #61520].
- Calling `Perl_sv_chop()` or otherwise upgrading an SV could result in an unaligned 64-bit access on the SPARC architecture [RT #60574].
- In the 5.10.0 release, `inc_version_list` would incorrectly list `5.10.*` after `5.8.*`; this affected the `@INC` search order [RT #67628].
- In 5.10.0, `pack "a*", $tainted_value` returned a non-tainted value [RT #52552].
- In 5.10.0, `printf` and `sprintf` could produce the fatal error `panic: utf8_mg_pos_cache_update` when printing UTF-8 strings [RT #62666].
- In the 5.10.0 release, a dynamically created `AUTOLOAD` method might be missed (method cache issue) [RT #60220,60232].
- In the 5.10.0 release, a combination of `use feature` and `//ee` could cause a memory leak [RT #63110].
- `-C` on the shebang (`#!`) line is once more permitted if it is also specified on the command line.

`-C` on the shebang line used to be a silent no-op *if* it was not also on the command line, so perl 5.10.0 disallowed it, which broke some scripts. Now perl checks whether it is also on the command line and only dies if it is not [RT #67880].

- In 5.10.0, certain types of re-entrant regular expression could crash, or cause the following assertion failure [RT #60508]:

```
Assertion rx->sublen >= (s - rx->subbeg) + i failed
```

- Perl now includes previously missing files from the Unicode Character Database.
- Perl now honors `TMPDIR` when opening an anonymous temporary file.

## Platform Specific Changes

Perl is incredibly portable. In general, if a platform has a C compiler, someone has ported Perl to it (or will soon). We're happy to announce that Perl 5.12 includes support for several new platforms. At the same time, it's time to bid farewell to some (very) old friends.

### New Platforms

Haiku

Perl's developers have merged patches from Haiku's maintainers. Perl should now build on Haiku.

MirOS BSD

Perl should now build on MirOS BSD.

### Discontinued Platforms

Domain/OS

MiNT

Tenon MachTen

### Updated Platforms

AIX

- Removed *libbsd* for AIX 5L and 6.1. Only `flock()` was used from *libbsd*.
- Removed *libgdbm* for AIX 5L and 6.1 if *libgdbm* < 1.8.3-5 is installed. The *libgdbm* is delivered as an optional package with the AIX Toolbox. Unfortunately the versions below 1.8.3-5 are broken.
- Hints changes mean that AIX 4.2 should work again.

Cygwin

- Perl now supports IPv6 on Cygwin 1.7 and newer.
- On Cygwin we now strip the last number from the DLL. This has been the behaviour in the cygwin.com build for years. The hints files have been updated.

Darwin (Mac OS X)

- Skip testing the be_BY.CP1131 locale on Darwin 10 (Mac OS X 10.6), as it's still buggy.
- Correct infelicities in the regexp used to identify buggy locales on Darwin 8 and 9 (Mac OS X 10.4 and 10.5, respectively).

DragonFly BSD

- Fix thread library selection [perl #69686]

FreeBSD

- The hints files now identify the correct threading libraries on FreeBSD 7 and later.

Irix

- We now work around a bizarre preprocessor bug in the Irix 6.5 compiler: `cc -E -` unfortunately goes into K&R mode, but `cc -E file.c` doesn't.

NetBSD

- Hints now supports versions 5.*.

OpenVMS

- `-UDEBUGGING` is now the default on VMS.

  Like it has been everywhere else for ages and ages. Also make command-line selection of -UDEBUGGING and -DDEBUGGING work in configure.com; before the only way to turn it off was by saying no in answer to the interactive question.

- The default pipe buffer size on VMS has been updated to 8192 on 64-bit systems.

- Reads from the in-memory temporary files of `PerlIO::scalar` used to fail if `$/` was set to a numeric reference (to indicate record-style reads). This is now fixed.

- VMS now supports `getgrgid`.

- Many improvements and cleanups have been made to the VMS file name handling and conversion code.

- Enabling the `PERL_VMS_POSIX_EXIT` logical name now encodes a POSIX exit status in a VMS condition value for better interaction with GNV's bash shell and other utilities that depend on POSIX exit values. See *"$?" in perlvms* for details.

- `File::Copy` now detects Unix compatibility mode on VMS.

Stratus VOS

- Various changes from Stratus have been merged in.

Symbian

- There is now support for Symbian S60 3.2 SDK and S60 5.0 SDK.

Windows

- Perl 5.12 supports Windows 2000 and later. The supporting code for legacy versions of Windows is still included, but will be removed during the next development cycle.

- Initial support for building Perl with MinGW-w64 is now available.

- *perl.exe* now includes a manifest resource to specify the `trustInfo` settings for Windows Vista and later. Without this setting Windows would treat *perl.exe* as a legacy application and apply various heuristics like redirecting access to protected file system areas (like the "Program Files" folder) to the users "VirtualStore" instead of generating a proper "permission denied" error.

  The manifest resource also requests the Microsoft Common-Controls version 6.0 (themed controls introduced in Windows XP). Check out the Win32::VisualStyles module on CPAN to switch back to old style unthemed controls for legacy applications.

- The `-t` filetest operator now only returns true if the filehandle is connected to a console window. In previous versions of Perl it would return true for all character mode devices, including *NUL* and *LPT1*.

- The `-p` filetest operator now works correctly, and the Fcntl::S_IFIFO constant is

defined when Perl is compiled with Microsoft Visual C. In previous Perl versions `-p` always returned a false value, and the Fcntl::S_IFIFO constant was not defined.

This bug is specific to Microsoft Visual C and never affected Perl binaries built with MinGW.

- The socket error codes are now more widely supported: The POSIX module will define the symbolic names, like POSIX::EWOULDBLOCK, and stringification of socket error codes in $! works as well now;

```
  C:\>perl -MPOSIX -E "$!=POSIX::EWOULDBLOCK; say $!"
  A non-blocking socket operation could not be completed
immediately.
```

- flock() will now set sensible error codes in $!. Previous Perl versions copied the value of $^E into $!, which caused much confusion.
- select() now supports all empty `fd_sets` more correctly.
- `'.\foo'` and `'..\foo'` were treated differently than `'./foo'` and `'../foo'` by `do` and `require` [RT #63492].
- Improved message window handling means that `alarm` and `kill` messages will no longer be dropped under race conditions.
- Various bits of Perl's build infrastructure are no longer converted to win32 line endings at release time. If this hurts you, please report the problem with the *perlbug* program included with perl.

## Known Problems

This is a list of some significant unfixed bugs, which are regressions from either 5.10.x or 5.8.x.

- Some CPANPLUS tests may fail if there is a functioning file *../../cpanp-run-perl* outside your build directory. The failure shouldn't imply there's a problem with the actual functional software. The bug is already fixed in [RT #74188] and is scheduled for inclusion in perl-v5.12.1.
- `List::Util::first` misbehaves in the presence of a lexical $_ (typically introduced by `my $_` or implicitly by `given`). The variable which gets set for each iteration is the package variable $_, not the lexical $_ [RT #67694].

  A similar issue may occur in other modules that provide functions which take a block as their first argument, like

```
foo { ... $_ ...} list
```

- Some regexes may run much more slowly when run in a child thread compared with the thread the pattern was compiled into [RT #55600].
- Things like `"\N{LATIN SMALL LIGATURE FF}" =~ /\N{LATIN SMALL LETTER F}+/` will appear to hang as they get into a very long running loop [RT #72998].
- Several porters have reported mysterious crashes when Perl's entire test suite is run after a build on certain Windows 2000 systems. When run by hand, the individual tests reportedly work fine.

## Errata

- This one is actually a change introduced in 5.10.0, but it was missed from that release's perldelta, so it is mentioned here instead.

  A bugfix related to the handling of the `/m` modifier and `qr` resulted in a change of behaviour

between 5.8.x and 5.10.0:

```
# matches in 5.8.x, doesn't match in 5.10.0
$re = qr/^bar/; "foo\nbar" =~ /$re/m;
```

## Acknowledgements

Perl 5.12.0 represents approximately two years of development since Perl 5.10.0 and contains over 750,000 lines of changes across over 3,000 files from over 200 authors and committers.

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 5.12.0:

Aaron Crane, Abe Timmerman, Abhijit Menon-Sen, Abigail, Adam Russell, Adriano Ferreira, Ã†var ArnfjÃ¶rÃ° Bjarmason, Alan Grover, Alexandr Ciornii, Alex Davies, Alex Vandiver, Andreas Koenig, Andrew Rodland, andrew@sundale.net, Andy Armstrong, Andy Dougherty, Jose AUGUSTE-ETIENNE, Benjamin Smith, Ben Morrow, bharanee rathna, Bo Borgerson, Bo Lindbergh, Brad Gilbert, Bram, Brendan O'Dea, brian d foy, Charles Bailey, Chip Salzenberg, Chris 'BinGOs' Williams, Christoph Lamprecht, Chris Williams, chromatic, Claes Jakobsson, Craig A. Berry, Dan Dascalescu, Daniel Frederick Crisman, Daniel M. Quinlan, Dan Jacobson, Dan Kogai, Dave Mitchell, Dave Rolsky, David Cantrell, David Dick, David Golden, David Mitchell, David M. Syzdek, David Nicol, David Wheeler, Dennis Kaarsemaker, Dintelmann, Peter, Dominic Dunlop, Dr.Ruud, Duke Leto, Enrico Sorcinelli, Eric Brine, Father Chrysostomos, Florian Ragwitz, Frank Wiegand, Gabor Szabo, Gene Sullivan, Geoffrey T. Dairiki, George Greer, Gerard Goossen, Gisle Aas, Goro Fuji, Graham Barr, Green, Paul, Hans Dieter Pearcey, Harmen, H. Merijn Brand, Hugo van der Sanden, Ian Goodacre, Igor Sutton, Ingo Weinhold, James Bence, James Mastros, Jan Dubois, Jari Aalto, Jarkko Hietaniemi, Jay Hannah, Jerry Hedden, Jesse Vincent, Jim Cromie, Jody Belka, John E. Malmberg, John Malmberg, John Peacock, John Peacock via RT, John P. Linderman, John Wright, Josh ben Jore, Jos I. Boumans, Karl Williamson, Kenichi Ishigaki, Ken Williams, Kevin Brintnall, Kevin Ryde, Kurt Starsinic, Leon Brocard, Lubomir Rintel, Luke Ross, Marcel GrÃ¼nauer, Marcus Holland-Moritz, Mark Jason Dominus, Marko Asplund, Martin Hasch, Mashrab Kuvatov, Matt Kraai, Matt S Trout, Max Maischein, Michael Breen, Michael Cartmell, Michael G Schwern, Michael Witten, Mike Giroux, Milosz Tanski, Moritz Lenz, Nicholas Clark, Nick Cleaton, Niko Tyni, Offer Kaye, Osvaldo Villalon, Paul Fenwick, Paul Gaborit, Paul Green, Paul Johnson, Paul Marquess, Philip Hazel, Philippe Bruhat, Rafael Garcia-Suarez, Rainer Tammer, Rajesh Mandalemula, Reini Urban, RenÃ©e BÃ¤cker, Ricardo Signes, Ricardo SIGNES, Richard Foley, Rich Rauenzahn, Rick Delaney, Risto Kankkunen, Robert May, Roberto C. Sanchez, Robin Barker, SADAHIRO Tomoyuki, Salvador Ortiz Garcia, Sam Vilain, Scott Lanning, SÃ©bastien Aperghis-Tramoni, SÃ©rgio Durigan JÃºnior, Shlomi Fish, Simon 'corecode' Schubert, Sisyphus, Slaven Rezic, Smylers, Steffen MÃ¼ller, Steffen Ullrich, Stepan Kasal, Steve Hay, Steven Schubiger, Steve Peters, Tels, The Doctor, Tim Bunce, Tim Jenness, Todd Rinaldo, Tom Christiansen, Tom Hukins, Tom Wyant, Tony Cook, Torsten Schoenfeld, Tye McQueen, Vadim Konovalov, Vincent Pit, Hio YAMASHINA, Yasuhiro Matsumoto, Yitzchak Scott-Thoennes, Yuval Kogman, Yves Orton, Zefram, Zsban Ambrus

This is woefully incomplete as it's automatically generated from version control history. In particular, it doesn't include the names of the (very much appreciated) contributors who reported issues in previous versions of Perl that helped make Perl 5.12.0 better. For a more complete list of all of Perl's historical contributors, please see the AUTHORS file in the Perl 5.12.0 distribution.

Our "retired" pumpkings Nicholas Clark and Rafael Garcia-Suarez deserve special thanks for their brilliant and substantive ongoing contributions. Nicholas personally authored over 30% of the patches since 5.10.0. Rafael comes in second in patch authorship with 11%, but is first by a long shot in committing patches authored by others, pushing 44% of the commits since 5.10.0 in this category, often after providing considerable coaching to the patch authors. These statistics in no way comprise all of their contributions, but express in shorthand that we couldn't have done it without them.

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 analyzed 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.

*http://dev.perl.org/perl5/errata.html* for a list of issues found after this release, as well as a list of CPAN modules known to be incompatible with this release.