NAME

perl5137delta - what is new for perl v5.13.7

DESCRIPTION

This document describes differences between the 5.13.6 release and the 5.13.7 release.

If you are upgrading from an earlier release such as 5.13.5, first read *perl5136delta*, which describes differences between 5.13.5 and 5.13.6.

Core Enhancements

Single term prototype

The + prototype is a special alternative to $ that will act like `\[@%]` when given a literal array or hash variable, but will otherwise force scalar context on the argument. This is useful for functions which should accept either a literal array or an array reference as the argument:

```
sub smartpush (+@) {
    my $aref = shift;
    die "Not an array or arrayref" unless ref $aref eq 'ARRAY';
    push @$aref, @_;
}
```

When using the + prototype, your function must check that the argument is of an acceptable type.

use re '/flags';

The `re` pragma now has the ability to turn on regular expression flags till the end of the lexical scope:

```
use re '/x';
"foo" =~ / (.+) /;  # /x implied
```

See *"'/flags' mode" in re* for details.

Statement labels can appear in more places

Statement labels can now occur before any type of statement or declaration, such as `package`.

use feature "unicode_strings" now applies to more regex matching

Another chunk of the *"The "Unicode Bug"" in perlunicode* is fixed in this release. Now, regular expressions compiled within the scope of the "unicode_strings" feature (or under the "u" regex modifier (specifiable currently only with infix notation `(?u:...)` or via `use re '/u'`) will match the same whether or not the target string is encoded in utf8, with regard to `[[:posix:]]` character classes

Work is underway to add the case sensitive matching to the control of this feature, but was not complete in time for this dot release.

Array and hash container functions accept references

All built-in functions that operate directly on array or hash containers now also accept hard references to arrays or hashes:

```
|----------------------------+----------------------------|
| Traditional syntax         | Terse syntax               |
|----------------------------+----------------------------|
| push @$arrayref, @stuff    | push $arrayref, @stuff     |
| unshift @$arrayref, @stuff | unshift $arrayref, @stuff  |
| pop @$arrayref             | pop $arrayref              |
| shift @$arrayref           | shift $arrayref            |
| splice @$arrayref, 0, 2    | splice $arrayref, 0, 2     |
```

```
| keys %$hashref             | keys $hashref             |
| keys @$arrayref            | keys $arrayref            |
| values %$hashref           | values $hashref           |
| values @$arrayref          | values $arrayref          |
| ($k,$v) = each %$hashref   | ($k,$v) = each $hashref   |
| ($k,$v) = each @$arrayref  | ($k,$v) = each $arrayref  |
|----------------------------+---------------------------|
```

This allows these built-in functions to act on long dereferencing chains or on the return value of subroutines without needing to wrap them in `@{}` or `%{}`:

```
push @{$obj->tags}, $new_tag;  # old way
push $obj->tags,    $new_tag;  # new way

for ( keys %{$hoh->{genres}{artists}} ) {...} # old way
for ( keys $hoh->{genres}{artists}    ) {...} # new way
```

For `push`, `unshift` and `splice`, the reference will auto-vivify if it is not defined, just as if it were wrapped with `@{}`.

Calling `keys` or `values` directly on a reference gives a substantial performance improvement over explicit dereferencing.

For `keys`, `values`, `each`, when overloaded dereferencing is present, the overloaded dereference is used instead of dereferencing the underlying reftype. Warnings are issued about assumptions made in the following three ambiguous cases:

```
(a) If both %{} and @{} overloading exists, %{} is used
(b) If %{} overloading exists on a blessed arrayref, %{} is used
(c) If @{} overloading exists on a blessed hashref, @{} is used
```

y///r

The `/r` flag, which was added to `s///` in 5.13.2, has been extended to the `y///` operator.

It causes it to perform the substitution on a *copy* of its operand, returning that copy instead of a character count.

New global variable ${^GLOBAL_PHASE}

A new global variable, `${^GLOBAL_PHASE}`, has been added to allow introspection of the current phase of the perl interpreter. It's explained in detail in *"${^GLOBAL_PHASE}" in perlvar* and *"BEGIN, UNITCHECK, CHECK, INIT and END" in perlmod*.

Unicode Version 6.0 is now supported (mostly)

Perl comes with the Unicode 6.0 data base updated with *Corrigendum #8*, with one exception noted below. See *http://unicode.org/versions/Unicode6.0.0* for details on the new release. Perl does not support any Unicode provisional properties, including the new ones for this release, but their database files are packaged with Perl.

Unicode 6.0 has chosen to use the name `BELL` for the character at U+1F514, which is a symbol that looks like a bell, and used in Japanese cell phones. This conflicts with the long-standing Perl usage of having `BELL` mean the ASCII `BEL` character, U+0007. In Perl 5.14, `\N{BELL}` will continue to mean U+0007, but its use will generate a deprecated warning message, unless such warnings are turned off. The new name for U+0007 in Perl will be `ALERT`, which corresponds nicely with the existing shorthand sequence for it, `"\a"`. `\N{BEL}` will mean U+0007, with no warning given. The character at U+1F514 will not have a name in 5.14, but can be referred to by `\N{U+1F514}`. The plan is that in Perl 5.16, `\N{BELL}` will refer to U+1F514, and so all code that uses `\N{BELL}` should convert by

then to using `\N{ALERT}`, `\N{BEL}`, or `"\a"` instead.

Improved support for custom OPs

Custom ops can now be registered with the new `custom_op_register` C function and the `XOP` structure. This will make it easier to add new properties of custom ops in the future. Two new properties have been added already, `xop_class` and `xop_peep`.

`xop_class` is one of the OA_*OP constants, and allows *B* and other introspection mechanisms to work with custom ops that aren't BASEOPs. `xop_peep` is a pointer to a function that will be called for ops of this type from `Perl_rpeep`.

See *"Custom Operators" in perlguts* and *"Custom Operators" in perlapi* for more detail.

The old `PL_custom_op_names`/`PL_custom_op_descs` interface is still supported but discouraged.

Incompatible Changes

Dereferencing typeglobs

If you assign a typeglob to a scalar variable:

```
$glob = *foo;
```

the glob that is copied to `$glob` is marked with a special flag indicating that the glob is just a copy. This allows subsequent assignments to `$glob` to overwrite the glob. The original glob, however, is immutable.

Many Perl operators did not distinguish between these two types of globs. This would result in strange behaviour in edge cases: `untie $scalar` would do nothing if the last thing assigned to the scalar was a glob (because it treated it as `untie *$scalar`, which unties a handle). Assignment to a glob slot (e.g., `(*$glob) = \@some_array`) would simply assign `\@some_array` to `$glob`.

To fix this, the `*{}` operator (including the `*foo` and `*$foo` forms) has been modified to make a new immutable glob if its operand is a glob copy. Various operators that make a distinction between globs and scalars have been modified to treat only immutable globs as globs.

This causes an incompatible change in code that assigns a glob to the return value of `*{}` when that operator was passed a glob copy. Take the following code, for instance:

```
$glob = *foo;
*$glob = *bar;
```

The `*$glob` on the second line returns a new immutable glob. That new glob is made an alias to `*bar`. Then it is discarded. So the second assignment has no effect.

It also means that `tie $handle` will now tie `$handle` as a scalar, even if it has had a glob assigned to it.

The upside to this incompatible change is that bugs *[perl #77496]*, *[perl #77502]*, *[perl #77508]*, *[perl #77688]*, and *[perl #77812]*, and maybe others, too, have been fixed.

See *http://rt.perl.org/rt3/Public/Bug/Display.html?id=77810* for even more detail.

Clearing stashes

Stash list assignment `%foo:: = ()` used to make the stash anonymous temporarily while it was being emptied. Consequently, any of its subroutines referenced elsewhere would become anonymous (showing up as "(unknown)" in `caller`). Now they retain their package names, such that `caller` will return the original sub name if there is still a reference to its typeglob, or "foo::__ANON__" otherwise *[perl #79208]*.

Deprecations

\N{BELL} is deprecated

This is because Unicode is using that name for a different character. See *Unicode Version 6.0 is now supported (mostly)* for more explanation.

Performance Enhancements

- When an object has many weak references to it, freeing that object can under some some circumstances take O(N^2) time to free (where N is the number of references). The number of circumstances has been reduced. *[perl #75254]*.

Modules and Pragmata

New Modules and Pragmata

- The following modules were added by the `Unicode::Collate` upgrade from 0.63 to 0.67. See below for details.

  `Unicode::Collate::CJK::Big5`

  `Unicode::Collate::CJK::GB2312`

  `Unicode::Collate::CJK::JISX0208`

  `Unicode::Collate::CJK::Korean`

  `Unicode::Collate::CJK::Pinyin`

  `Unicode::Collate::CJK::Stroke`

Updated Modules and Pragmata

- `Archive::Extract` has been upgraded from 0.44 to 0.46

  Resolves an issue with NetBSD-current and its new unzip executable.

- `Archive::Tar` has been upgraded from 1.68 to 1.72

  This adds the ptargrep utility for using regular expressions against the contents of files in a tar archive.

- `B` has been upgraded from 1.24 to 1.26.

  It no longer crashes when taking apart a `y///` containing characters outside the octet range or compiled in a `use utf8` scope.

  The size of the shared object has been reduced by about 40%, with no reduction in functionality.

- `B::Deparse` has been upgraded from 0.99 to 1.01.

  It fixes deparsing of `our` followed by a variable with funny characters (as permitted under the `utf8` pragma) *[perl #33752]*.

- `CGI` has been upgraded from 3.49 to 3.50

  This provides the following security fixes: the MIME boundary in multipart_init is now random and improvements to the handling of newlines embedded in header values.

  The documentation for param_fetch() has been corrected and clarified.

- `CPAN` has been upgraded from 1.94_61 to 1.94_62

- `CPANPLUS` has been upgraded from 0.9007 to 0.9010

  Fixes for the SQLite source engine and resolving of issues with the testsuite when run under local::lib and/or cpanminus

- `CPANPLUS::Dist::Build` has been upgraded from 0.48 to 0.50

- `Data::Dumper` has been upgraded from 2.129 to 2.130_01.

- DynaLoader has been upgraded from 1.10 to 1.11.

  It fixes a buffer overflow when passed a very long file name.

- ExtUtils::Constant has been upgraded from 0.22 to 0.23.

  The AUTOLOAD helper code generated by ExtUtils::Constant::ProxySubs can now croak for missing constants, or generate a complete AUTOLOAD subroutine in XS, allowing simplification of many modules that use it. (Fcntl, File::Glob, GDBM_File, I18N::Langinfo, POSIX, Socket)

  ExtUtils::Constant::ProxySubs can now optionally push the names of all constants onto the package's C{@EXPORT_OK}. This has been used to replace less space-efficient code in B, helping considerably shrink the size of its shared object.

- Fcntl has been upgraded from 1.09 to 1.10.

- File::Fetch has been upgraded from 0.24 to 0.28

  HTTP::Lite is now supported for 'http' scheme.

  The fetch utility is supported on FreeBSD, NetBSD and Dragonfly BSD for the http and ftp schemes.

- File::Glob has been upgraded from 1.09 to 1.10.

- File::stat has been upgraded from 1.03 to 1.04.

  The -x and -X file test operators now work correctly under the root user.

- GDBM_File has been upgraded from 1.11 to 1.12.

  This fixes a memory leak when DBM filters are used.

- Hash::Util has been upgraded from 0.09 to 0.10.

- Hash::Util::FieldHash has been upgraded from 1.05 to 1.06.

- I18N::Langinfo has been upgraded from 0.06 to 0.07.

- Locale::Maketext has been upgraded from 1.16 to 1.17.

- Math::BigInt has been upgraded from 1.97 to 1.99_01.

- Math::BigRat has been upgraded from 0.26 to 0.26_01

- Math::BigInt::FastCalc has been upgraded from 0.22 to 0.24_01.

- MIME::Base64 has been upgraded from 3.09 to 3.10

  Includes new functions to calculate the length of encoded and decoded base64 strings.

- mro has been upgraded from 1.04 to 1.05.

- NDBM_File has been upgraded from 1.09 to 1.10.

  This fixes a memory leak when DBM filters are used.

- ODBM_File has been upgraded from 1.08 to 1.09.

  This fixes a memory leak when DBM filters are used.

- Opcode has been upgraded from 1.16 to 1.17.

- parent has been upgraded from 0.223 to 0.224

- Pod::Simple has been upgraded from 3.14 to 3.15

  Includes various fixes to HTML and XHTML handling.

- `POSIX` has been upgraded from 1.21 to 1.22.
- `re` has been upgraded from 0.13 to 0.14, for the sake of the new `use re "/flags"` pragma.
- `Safe` has been upgraded from 2.28 to 2.29.

  It adds `&version::vxs::VCMP` to the default share.
- `SDBM_File` has been upgraded from 1.07 to 1.08.
- `SelfLoader` has been upgraded from 1.17 to 1.18.

  It now works in taint mode *[perl #72062]*.
- `Socket` has been upgraded from 1.90 to 1.91.
- `Storable` has been upgraded from 2.22 to 2.24

  Includes performance improvement for overloaded classes.
- `Sys::Hostname` has been upgraded from 1.13 to 1.14.
- `Unicode::Collate` has been upgraded from 0.63 to 0.67

  This release newly adds locales `ja` `ko` and `zh` and its variants (`zh__big5han`, `zh__gb2312han`, `zh__pinyin`, `zh__stroke`).

  Supported UCA_Version 22 for Unicode 6.0.0.

  The following modules have been added:

  `Unicode::Collate::CJK::Big5` for `zh__big5han` which makes tailoring of CJK Unified Ideographs in the order of CLDR's big5han ordering.

  `Unicode::Collate::CJK::GB2312` for `zh__gb2312han` which makes tailoring of CJK Unified Ideographs in the order of CLDR's gb2312han ordering.

  `Unicode::Collate::CJK::JISX0208` which makes tailoring of 6355 kanji (CJK Unified Ideographs) in the JIS X 0208 order.

  `Unicode::Collate::CJK::Korean` which makes tailoring of CJK Unified Ideographs in the order of CLDR's Korean ordering.

  `Unicode::Collate::CJK::Pinyin` for `zh__pinyin` which makes tailoring of CJK Unified Ideographs in the order of CLDR's pinyin ordering.

  `Unicode::Collate::CJK::Stroke` for `zh__stroke` which makes tailoring of CJK Unified Ideographs in the order of CLDR's stroke ordering.

Documentation

perlvar reorders the variables and groups them by topic. Each variable introduced after Perl 5.000 notes the first version in which it is available. *perlvar* also has a new section for deprecated variables to note when they were removed.

New Documentation

perlpodstyle

New style guide for POD documentation, split mostly from the NOTES section of the pod2man man page.

(This was added to `v5.13.6` but was not documented with that release).

Changes to Existing Documentation

- Array and hash slices in scalar context are now documented in *perldata*.
- *perlform* and *perllocale* have been corrected to state that `use locale` affects formats.

Diagnostics

New Diagnostics

- "Using !~ with %s doesn't make sense": This message was actually added in 5.13.2, but was omitted from perldelta. It now applies also to the `y///` operator, and has been documented.

Utility Changes

ptargrep

- *ptargrep* is a utility to apply pattern matching to the contents of files in a tar archive. It comes with `Archive::Tar`.

Testing

- The new *t/mro/isa_aliases.t* has been added, which tests that `*Foo::ISA = *Bar::ISA` works properly.
- *t/mro/isarev.t* has been added, which tests that `PL_isarev` (accessible at the Perl level via `mro::get_isarev`) is updated properly.
- *t/run/switchd-78586.t* has been added, which tests that *[perl #78586]* has been fixed (related to line numbers in the debugger).

Platform Support

Platform-Specific Notes

Windows

Directory handles are now properly cloned when threads are created. In perl 5.13.6, child threads simply stopped inheriting directory handles. In previous versions, threads would share handles, resulting in crashes.

Support for building with Visual C++ 2010 is now underway, but is not yet complete. See *README.win32* for more details.

VMS

Record-oriented files (record format variable or variable with fixed control) opened for write by the perlio layer will now be line buffered to prevent the introduction of spurious line breaks whenever the perlio buffer fills up.

Internal Changes

- `lex_start` has been added to the API, but is considered experimental.
- A new `parse_block` function has been added to the API *[perl #78222]*.
- A new, experimental API has been added for accessing the internal structure that Perl uses for `%^H`. See the functions beginning with `cophh_` in *perlapi*.
- A stash can now have a list of effective names in addition to its usual name. The first effective name can be accessed via the `HvENAME` macro, which is now the recommended name to use in MRO linearisations (`HvNAME` being a fallback if there is no `HvENAME`).

  These names are added and deleted via `hv_ename_add` and `hv_ename_delete`. These two functions are *not* part of the API.
- The way the parser handles labels has been cleaned up and refactored. As a result, the `newFOROP()` constructor function no longer takes a parameter stating what label is to go in the state op.
- The `newWHILEOP()` and `newFOROP()` functions no longer accept a line number as a parameter.
- A new `parse_barestmt()` function has been added, for parsing a statement without a label.

- A new `parse_label()` function has been added, that parses a statement label, separate from statements.
- The `CvSTASH()` macro can now only be used as an rvalue. `CvSTASH_set()` has been added to replace assignment to `CvSTASH()`. This is to ensure that backreferences are handled properly. These macros are not part of the API.
- The `op_scope()` and `op_lvalue()` functions have been added to the API, but are considered experimental.

Selected Bug Fixes

- The `parse_stmt` C function added in earlier in the 5.13.x series has been fixed to work with statements ending with `}` *[perl #78222]*.
- The `parse_fullstmt` C function added in 5.13.5 has been fixed to work when called while an expression is being parsed.
- Characters in the Latin-1 non-ASCII range (0x80 to 0xFF) used not to match themselves if the string happened to be UTF8-encoded internally, the regular expression was not, and the character in the regular expression was inside a repeated group (e.g., `Encode::decode_utf8("\303\200") =~ /(\xc0)+/`) *[perl #78464]*.
- The `(?d)` regular expression construct now overrides a previous `(?u)` or `use feature "unicode_string"` *[perl #78508]*.
- A memory leak in `do "file"`, introduced in perl 5.13.6, has been fixed *[perl #78488]*.
- Various bugs related to typeglob dereferencing have been fixed. See *Dereferencing typeglobs* , above.
- The `SvPVbyte` function available to XS modules now calls magic before downgrading the SV, to avoid warnings about wide characters *[perl #72398]*.
- The = operator used to ignore magic (e.g., tie methods) on its right-hand side if the scalar happened to hold a typeglob. This could happen if a typeglob was the last thing returned from or assigned to a tied scalar *[perl #77498]*.
- `sprintf` was ignoring locales when called with constant arguments *[perl #78632]*.
- A non-ASCII character in the Latin-1 range could match both a Posix class, such as `[[:alnum:]]`, and its inverse `[[:^alnum:]]`. This is now fixed for regular expressions compiled under the `"u"` modifier. See *`use feature "unicode_strings"` now applies to more regex matching*. *[perl #18281]*.
- Concatenating long strings under `use encoding` no longer causes perl to crash *[perl #78674]*.
- Typeglob assignments would crash if the glob's stash no longer existed, so long as the glob assigned to was named 'ISA' or the glob on either side of the assignment contained a subroutine.
- Calling `->import` on a class lacking an import method could corrupt the stack, resulting in strange behaviour. For instance,

  ```
  push @a, "foo", $b = bar->import;
  ```

  would assign 'foo' to $b *[perl #63790]*.
- Creating an alias to a package when that package had been detached from the symbol table would result in corrupted isa caches *[perl #77358]*.
- `.=` followed by <> or `readline` would leak memory if `$/` contained characters beyond the

octet range and the scalar assigned to happened to be encoded as UTF8 internally *[perl #72246]*.

- The `recv` function could crash when called with the MSG_TRUNC flag *[perl #75082]*.
- Evaluating a simple glob (like `*a`) was calling get-magic on the glob, even when its contents were not being used *[perl #78580]*.

  This bug was introduced in 5.13.2 and did not affect earlier perl versions.
- Matching a Unicode character against an alternation containing characters that happened to match continuation bytes in the former's UTF8 representation (`qq{\x{30ab}} =~ /\xab|\xa9/`) would cause erroneous warnings *[perl #70998]*.
- `s///r` (added in 5.13.2) no longer leaks.
- The trie optimisation was not taking empty groups into account, preventing 'foo' from matching `/\A(?:(?:)foo|bar|zot)\z/` *[perl #78356]*.
- A pattern containing a + inside a lookahead would sometimes cause an incorrect match failure in a global match (e.g., `/(?=(\S+))/g`) *[perl #68564]*.
- Iterating with `foreach` over an array returned by an lvalue sub now works *[perl #23790]*.
- $@ is now localised during calls to `binmode` to prevent action at a distance *[perl #78844]*.
- `PL_isarev`, which is accessible to Perl via `mro::get_isarev` is now updated properly when packages are deleted or removed from the `@ISA` of other classes. This allows many packages to be created and deleted without causing a memory leak *[perl #75176]*.
- `undef *Foo::` and `undef *Foo::ISA` and `delete $package::{ISA}` used not to update the internal isa caches if the stash or `@ISA` array had a reference elsewhere. In fact, `undef *Foo::ISA` would stop a new `@Foo::ISA` array from updating caches.
- `@ISA` arrays can now be shared between classes via `*Foo::ISA = \@Bar::ISA` or `*Foo::ISA = *Bar::ISA` *[perl #77238]*.
- The parser no longer hangs when encountering certain Unicode characters, such as U+387 *[perl #74022]*.
- `formline` no longer crashes when passed a tainted format picture. It also taints $^A now if its arguments are tainted *[perl #79138]*.
- A signal handler called within a signal handler could cause leaks or double-frees. Now fixed. *[perl #76248]*.
- When trying to report `Use of uninitialized value $Foo::BAR`, crashes could occur if the GLOB of the global variable causing the warning has been detached from its original stash by, for example `delete $::{'Foo::'}`. This has been fixed by disabling the reporting of variable names in the warning in those cases.

Obituary

Randy Kobes, creator of the kobesearch alternative to search.cpan.org and contributor/maintainer to several core Perl toolchain modules, passed away on September 18, 2010 after a battle with lung cancer. His contributions to the Perl community will be missed.

Acknowledgements

Perl 5.13.7 represents approximately one month of development since Perl 5.13.6 and contains 73100 lines of changes across 518 files from 39 authors and committers:

Abhijit Menon-Sen, Abigail, Ben Morrow, Chas. J. Owens IV, Chris 'BinGOs' Williams, Craig A. Berry, David Golden, David Mitchell, Father Chrysostomos, Fingle Nark, Florian Ragwitz, George Greer,

Grant McLean, H.Merijn Brand, Ian Goodacre, Jan Dubois, Jerry D. Hedden, Jesse Vincent, Karl Williamson, Lubomir Rintel, Marty Pauley, Moritz Lenz, Nicholas Clark, Nicolas Kaiser, Niko Tyni, Peter John Acklam, Rafael Garcia-Suarez, Shlomi Fish, Steffen Mueller, Steve Hay, Tatsuhiko Miyagawa, Tim Bunce, Todd Rinaldo, Tom Christiansen, Tom Hukins, Tony Cook, Yves Orton, Zefram and brian d foy

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

Reporting Bugs

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

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

If the bug you are reporting has security implications, which make it inappropriate to send to a publicly archived mailing list, then please send it to perl5-security-report@perl.org. This points to a closed subscription unarchived mailing list, which includes all the core committers, who be able to help assess the impact of issues, figure out a resolution, and help co-ordinate the release of patches to mitigate or fix the problem across all platforms on which Perl is supported. Please only use this address for security issues in the Perl core, not for modules independently distributed on CPAN.

SEE ALSO

The *Changes* file for an explanation of how to view exhaustive details on what changed.

The *INSTALL* file for how to build Perl.

The *README* file for general stuff.

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