co_flags.CO_NOFREE is now always set correctly by the code object constructor based on freevars and cellvars, rather than needing to be set correctly by the caller. This ensures it will be cleared automatically when additional cell references are injected into a modified code object and function.

Fixed several issues in printing tracebacks (PyTraceBack_Print()). Setting sys.tracebacklimit to 0 or less now suppresses printing tracebacks. Setting sys.tracebacklimit to None now causes using the default limit. Setting sys.tracebacklimit to an integer larger than LONG_MAX now means using the limit LONG_MAX rather than the default limit. Fixed integer overflows in the case of more than `2**31` traceback items on Windows. Fixed output errors handling.

Fix the interactive interpreter looping endlessly when no memory.

Bytearray methods partition() and rpartition() now accept only bytes-like objects as separator, as documented. In particular they now raise TypeError rather of returning a bogus result when an integer is passed as a separator.

Fix a segmentation fault caused by a combination of the async soft keyword and continuation lines.

BytesWarning no longer emitted when the *fromlist* argument of `__import__()` or the `__all__` attribute of the module contain bytes instances.

Fixed OverflowError in the 'unicode-escape' codec and in codecs.escape_decode() when decode an escaped non-ascii byte.

Print the full context/cause chain of exceptions on interpreter exit, even if an exception in the chain is unhashable or compares equal to later ones. Patch by Zane Bitter.

Fix timeout rounding in the select module to round correctly negative timeouts between -1.0 and 0.0. The functions now block waiting for events as expected. Previously, the call was incorrectly non-blocking. Patch by Pablo Galindo.

Restored blocking "from package import module" by setting sys.modules["package.module"] to None.

Fixed a bug in debug memory allocator. There was a write to freed memory after shrinking a memory block.

Fixed a ValueError when convert a string with large number of underscores to integer with binary base.

Fixed an assertion failure in Python parser in case of a bad *unicodedata.normalize()*. Patch by Oren Milman.

Raise a *TypeError* with a helpful error message when class creation fails due to a metaclass with a bad `__prepare__()` method. Patch by Oren Milman.

Fix an assertion failure in *_warnings.warn()* in case of a bad `__name__` global. Patch by Oren Milman.

Fix an assertion failure in *json*, in case *_json.make_encoder()* received a bad *encoder()* argument. Patch by Oren Milman.

Fix assertion failures in case of failing to import from a module with a bad `__name__` attribute, and in case of failing to access an attribute of such a module. Patch by Oren Milman.

Fix an assertion failure in *ctypes* class definition, in case the class has an attribute whose name is specified in `_anonymous_` but not in `_fields_`. Patch by Oren Milman.

Fix an assertion failure in *_random.Random.seed()* in case the argument has a bad `__abs__()` method. Patch by Oren Milman.

Fix an assertion failure in imp.create_dynamic(), when spec.name is not a string. Patch by Oren Milman.

Fix a crash in the `__setstate__()` method of *ctypes._CData*, in case of a bad `__dict__`. Patch by Oren Milman.

Fix crashes in true division and multiplication of a timedelta object by a float with a bad as_integer_ratio() method. Patch by Oren Milman.

Fix an assertion failure in *warnings.warn_explicit*, when the return value of the received loader's get_source() has a bad splitlines() method. Patch by Oren Milman.

PyErr_PrintEx() clears now the ignored exception that may be raised by *_PySys_SetObjectId()*, for example when no memory.

Two minor fixes for `typing` module: allow shallow copying instances of generic classes, improve interaction of `__init_subclass__` with generics. Original PRs by Ivan Levkivskyi.

The header folding algorithm for the new email policies has been rewritten, which also fixes bpo-30788, bpo-31831, and bpo-32182. In particular, RFC2231 folding is now done correctly.

io.FileIO.readall() and io.FileIO.read() now release the GIL when getting the file size. Fixed hang of all threads with inaccessible NFS server. Patch by Nir Soffer.

Make :meth:`msilib.SummaryInformation.GetProperty` return `None` when the value of property is `VT_EMPTY`. Initial patch by Mark Mc Mahon.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\(cpython-main)(Misc)(NEWS.d)3.6.4rc1.rst`, line 297); *backlink***
>
> Unknown interpreted text role "meth".

Fix wrong usage of :func:`collections.namedtuple` in the :meth:`RobotFileParser.parse() <urllib.robotparser.RobotFileParser.parse>` method. Initial patch by Robin Wellner.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\(cpython-main)(Misc)(NEWS.d)3.6.4rc1.rst, line 307); *backlink*

Unknown interpreted text role "func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\(cpython-main)(Misc)(NEWS.d)3.6.4rc1.rst, line 307); *backlink*

Unknown interpreted text role "meth".

:func:`msilib.OpenDatabase` now raises a better exception message when it couldn't open or create an MSI file. Initial patch by William TisÃ¤ter.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\(cpython-main)(Misc)(NEWS.d)3.6.4rc1.rst, line 319); *backlink*

Unknown interpreted text role "func".

`codecs.StreamReader.read(n)` now returns not more than *n* characters/bytes for non-negative *n*. This makes it compatible with `read()` methods of other file-like objects.

Fixed issues with binary plists: Fixed saving bytearrays. Identical objects will be saved only once. Equal references will be load as identical objects. Added support for saving and loading recursive data structures.

Make asyncio.IncompleteReadError and LimitOverrunError pickleable.

Fixed the looping of asyncio in the case of reconnection the socket during waiting async read/write from/to the socket.

Restored support of loading marshal files with the TYPE_INT64 code. These files can be produced in Python 2.7.

Reduce performance overhead of asyncio debug mode.

Fixed determining the MAC address in the uuid module: Using ifconfig on NetBSD and OpenBSD. Using arp on Linux, FreeBSD, NetBSD and OpenBSD. Based on patch by Takayuki Shimizukawa.

Fix potential missed signal in signal.signal().

Fix Blake2 params leaf_size and node_offset on big endian platforms. Patch by Jack O'Connor.

Fixed compilation of the socket module on NetBSD 8. Fixed assertion failure or reading arbitrary data when parse a AF_BLUETOOTH address on NetBSD and DragonFly BSD.

Fixed stack corruption in curses.box() and curses.ungetmouse() when the size of types chtype or mmask_t is less than the size of C long. curses.box() now accepts characters as arguments. Based on patch by Steve Fink.

plistlib now catches more errors when read binary plists and raises InvalidFileException instead of unexpected exceptions.

Fix the method for checking pad state of curses WINDOW. Patch by Masayuki Yamamoto.

Fixed the layout of the kqueue_event structure on OpenBSD and NetBSD. Fixed the comparison of the kqueue_event objects.

Fixed building the curses module on NetBSD.

Instances of pickle.Pickler subclass with the persistent_id() method and pickle.Unpickler subclass with the persistent_load() method no longer create reference cycles.

Fix multiprocessing.Process when stdout and/or stderr is closed or None.

If nested log adapters are used, the inner `process()` methods are no longer omitted.

The `manager` property on LoggerAdapter objects is now properly settable.

Fix timeout rounding in time.sleep(), threading.Lock.acquire() and socket.socket.settimeout() to round correctly negative timeouts between -1.0 and 0.0. The functions now block waiting for events as expected. Previously, the call was incorrectly non-blocking. Patch by Pablo Galindo.

traceback: Fix a TypeError that occurred during printing of exception tracebacks when either the current exception or an exception in its context/cause chain is unhashable. Patch by Zane Bitter.

Fixed buffer overflow in select.kqueue.control().

Prevent a crash when calling the `__init__()` method of a `sqlite3.Cursor` object more than once. Patch by Oren Milman.

`idpattern` in `string.Template` matched some non-ASCII characters. Now it uses `-i` regular expression local flag to avoid non-ASCII characters.

Prevent a crash in `sqlite3.Cursor.close()` in case the `Cursor` object is uninitialized. Patch by Oren Milman.

Fix possible crash in timedelta constructor called with custom integers.

On Windows, faulthandler.enable() now ignores MSC and COM exceptions.

Prevent crashes in *_elementtree* due to unsafe cleanup of *Element.text* and *Element.tail*. Patch by Oren Milman.

an empty asyncio.Queue now doesn't leak memory when queue.get pollers timeout

Fix method set_protocol() of class _SSLProtocolTransport in asyncio module. This method was previously modifying a wrong reference to the protocol.

Fixed memory leaks in Tkinter's methods splitlist() and split() when pass a string larger than 2 GiB.

Fixed typo in the name of Tkinter's method adderrorinfo().

Fix the string representation of a netrc object.

Added a workaround for getkey() in curses for ncurses 5.7 and earlier.

Avoid venv activate failures with undefined variables

inspect.unwrap() will now only try to unwrap an object sys.getrecursionlimit() times, to protect against objects which create a new object on every attribute access.

Stop crashes when concurrently iterate over itertools.groupby() iterators.

`threading.current_thread()` should not return a dummy thread at shutdown.

python -m ensurepip now exits with non-zero exit code if pip bootstrapping has failed.

`random.seed()` now works with bytes in version=1

Fix `poll.poll([timeout])` in the `select` module for arbitrary negative timeouts on all OSes where it can only be a non-negative integer or -1. Patch by Riccardo Coccioli.

multiprocessing's semaphore tracker should be launched again if crashed.

Make multiprocessing's forkserver process immune to Ctrl-C and other user interruptions. If it crashes, restart it when necessary.

Added asyncio.BaseEventLoop.connect_accepted_socket versionadded marker.

Fix incorrect usage of `get_history_length` in readline documentation example code. Patch by Brad Smith.

The operator functions without double underscores are preferred for clarity. The one with underscores are only kept for back-compatibility.

Skip test_httpservers test_undecodable_file on macOS: fails on APFS.

Skip test_socket.test_sha256() on Linux kernel older than 4.5. The test fails with ENOKEY on kernel 3.10 (on ppc64le). A fix was merged into the kernel 4.5.

Fix test_tools.test_unparse: DirectoryTestCase now stores the names sample to always test the same files. It prevents false alarms when hunting reference leaks.

Add the *set_nomemory(start, stop)* and *remove_mem_hooks()* functions to the _testcapi module.

`detect_modules()` in `setup.py` now also searches the sysroot paths when cross-compiling.

Fixes Windows SDK version detection when building for Windows.

Fixes quotes in PCbuild/clean.bat

Abort the build when building out of a not clean source tree.

Fixed Argument Clinic sometimes causing compilation errors when there was more than one function and/or method in a .c file with the same name.

Update Windows builds to use SQLite 3.21.0.

Update OS X installer to use SQLite 3.21.0.

Prevent double substitution of prefix in python-config.sh.

Avoid wholesale rebuild after *make regen-all* if nothing changed.

Return `None` when `View.Fetch()` returns `ERROR_NO_MORE_ITEMS` instead of raising `MSIError`. Initial patch by Anthony Tuininga.

Fixes Modify button in Apps and Features dialog.

Update macOS installer to use OpenSSL 1.0.2m

Improve tk event exception tracebacks in IDLE. When tk event handling is driven by IDLE's run loop, a confusing and distracting queue.EMPTY traceback context is no longer added to tk event exception tracebacks. The traceback is now the same as when event handling is driven by user code. Patch based on a suggestion by Serhiy Storchaka.

Delete unused file idlelib/tabbedpages.py. Use of TabbedPageSet in configdialog was replaced by ttk.Notebook.

IDLE: Fix old and new bugs in pathbrowser; improve tests. Patch mostly by Cheryl Sabella.

IDLE -- Restrict shell prompt manipulation to the shell. Editor and output windows only see an empty last prompt line. This simplifies

the code and fixes a minor bug when newline is inserted. Sys.ps1, if present, is read on Shell start-up, but is not set or changed.

The font sample in the IDLE configuration dialog is now editable. Changes persist while IDLE remains open

Test_code_module now passes if run after test_idle, which sets ps1. The code module uses sys.ps1 if present or sets it to '>>> ' if not. Test_code_module now properly tests both behaviors. Ditto for ps2.

Fix a TypeError that caused a shell restart when printing a traceback that includes an exception that is unhashable. Patch by Zane Bitter.

Use non-Latin characters in the IDLE's Font settings sample. Even if one selects a font that defines a limited subset of the unicode Basic Multilingual Plane, tcl/tk will use other fonts that define a character. The expanded example give users of non-Latin characters a better idea of what they might see in IDLE's shell and editors. To make room for the expanded sample, frames on the Font tab are re-arranged. The Font/Tabs help explains a bit about the additions.

Simplify the API of IDLE's Module Browser. Passing a widget instead of an flist with a root widget opens the option of creating a browser frame that is only part of a window. Passing a full file name instead of pieces assumed to come from a .py file opens the possibility of browsing python files that do not end in .py.

IDLE - Make _htest, _utest parameters keyword only.

Remove test order dependence in idle_test.test_browser.

Rename IDLE's module browser from Class Browser to Module Browser. The original module-level class and method browser became a module browser, with the addition of module-level functions, years ago. Nested classes and functions were added yesterday. For back-compatibility, the virtual event <<open-class-browser>>, which appears on the Keys tab of the Settings dialog, is not changed. Patch by Cheryl Sabella.

Default fonts now are scaled on HiDPI displays.

IDLE module browser now shows nested classes and functions. Original patches for code and tests by Guilherme Polo and Cheryl Sabella, respectively.

Make redemo work with Python 3.6 and newer versions. Also, remove the `LOCALE` option since it doesn't work with string patterns in Python 3. Patch by Christoph Sarnowski.

Fix PyGILState_Ensure(). When PyGILState_Ensure() is called in a non-Python thread before PyEval_InitThreads(), only call PyEval_InitThreads() after calling PyThreadState_New() to fix a crash.

Fix memory corruption due to allocator mix in getpath.c between Py_GetPath() and Py_SetPath()

The *PyExc_RecursionErrorInst* singleton is removed and *PyErr_NormalizeException()* does not use it anymore. This singleton is persistent and its members being never cleared may cause a segfault during finalization of the interpreter. See also issue #22898.