SSLObject.version() now correctly returns None when handshake over BIO has not been performed yet.

Add fuzz tests for float(str), int(str), unicode(str); for oss-fuzz.

Upgrade libexpat embedded copy from version 2.2.1 to 2.2.3 to get security fixes.

Prevent environment variables injection in subprocess on Windows. Prevent passing other environment variables and command arguments.

Upgrade expat copy from 2.2.0 to 2.2.1 to get fixes of multiple security vulnerabilities including: CVE-2017-9233 (External entity infinite loop DoS), CVE-2016-9063 (Integer overflow, re-fix), CVE-2016-0718 (Fix regression bugs from 2.2.0's fix to CVE-2016-0718) and CVE-2012-0876 (Counter hash flooding with SipHash). Note: the CVE-2016-5300 (Use os-specific entropy sources like getrandom) doesn't impact Python, since Python already gets entropy from the OS to set the expat secret using `XML_SetHashSalt()`.

Fix urllib.parse.splithost() to correctly parse fragments. For example, `splithost('//127.0.0.1#@evil.com/')` now correctly returns the `127.0.0.1` host, instead of treating `@evil.com` as the host in an authentication (`login@host`).

Update expat copy from 2.1.1 to 2.2.0 to get fixes of CVE-2016-0718 and CVE-2016-4472. See https://sourceforge.net/p/expat/bugs/537/ for more information.

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 *subprocess.Popen()* on Windows, in case the env argument has a bad keys() method. Patch by Oren Milman.

Fix an assertion failure in *PyErr_WriteUnraisable()* in case of an exception with a bad `__module__` attribute. Patch by Oren Milman.

Fix assertion failures in case of a bad warnings.filters or warnings.defaultaction. Patch by Oren Milman.

Change direct usage of PyInterpreterState.modules to PyImport_GetModuleDict(). Also introduce more uniformity in other code that deals with sys.modules. This helps reduce complications when working on sys.modules.

Switch to the abstract API when dealing with `PyInterpreterState.modules`. This allows later support for all dict subclasses and other Mapping implementations. Also add a `PyImport_GetModule()` function to reduce a bunch of duplicated code.

Raise a TypeError instead of SystemError in case warnings.onceregistry is not a dictionary. Patch by Oren Milman.

For finer control of tracing behaviour when testing the interpreter, two new frame attributes have been added to control the emission of particular trace events: `f_trace_lines` (`True` by default) to turn off per-line trace events; and `f_trace_opcodes` (`False` by default) to turn on per-opcode trace events.

Fix several possible instances of undefined behavior due to floating-point demotions.

Location information (`lineno` and `col_offset`) in f-strings is now (mostly) correct. This fixes tools like flake8 from showing warnings on the wrong line (typically the first line of the file).

Consolidate CPython's global runtime state under a single struct. This improves discoverability of the runtime state.

Fix possible undefined behavior in _PyObject_FastCall_Prepend.

Include sys/sysmacros.h for major(), minor(), and makedev(). GNU C libray plans to remove the functions from sys/types.h.

Fix an assertion failure in *zipimport.zipimporter.get_data* on Windows, when the return value of `pathname.replace('/','\\')` isn't a string. Patch by Oren Milman.

Fix an assertion failure in the write() method of *io.TextIOWrapper*, when the encoder doesn't return a bytes object. Patch by Oren Milman.

Fix a crash in some methods of *io.TextIOWrapper*, when the decoder's state is invalid. Patch by Oren Milman.

`print` now shows correct usage hint for using Python 2 redirection syntax. Patch by Sanyam Khurana.

Fix a race condition in importlib _get_module_lock().

Add a non-dummy implementation of _Py_atomic_store and _Py_atomic_load on MSVC.

Fix potential crash during GC caused by `tp_dealloc` which doesn't call `PyObject_GC_UnTrack()`.

Avoid masking original TypeError in call with * unpacking when other arguments are passed.

str.format_map() now passes key lookup exceptions through. Previously any exception was replaced with a KeyError exception.

Use _Py_atomic API for concurrency-sensitive signal state.

Relative import from unloaded package now reimports the package instead of failing with SystemError. Relative import from non-package now fails with ImportError rather than SystemError.

Improve signal delivery. Avoid using Py_AddPendingCall from signal handler, to avoid calling signal-unsafe functions. The tests I'm adding here fail without the rest of the patch, on Linux and OS X. This means our signal delivery logic had defects (some signals could be lost).

Avoid blocking in pthread_mutex_lock() when PyThread_acquire_lock() is asked not to block.

Make sure the 'Missing parentheses' syntax error message is only applied to SyntaxError, not to subclasses. Patch by Martijn Pieters.

Fixed a race condition when import a submodule from a package.

The internal unicodedata database has been upgraded to Unicode 10.0.

Move co_extra_freefuncs from per-thread to per-interpreter to avoid crashes.

`print` now shows expected input in custom error message when used as a Python 2 statement. Patch by Sanyam Khurana.

Removed a too-strict assertion that failed for certain f-strings, such as eval("f'\n'") and eval("f'\r'").

The compiler now produces more optimal code for complex condition expressions in the "if", "while" and "assert" statement, the "if" expression, and generator expressions and comprehensions.

Implement PEP 538 (legacy C locale coercion). This means that when a suitable coercion target locale is available, both the core interpreter and locale-aware C extensions will assume the use of UTF-8 as the default text encoding, rather than ASCII.

Allows setting cell values for __closure__. Patch by Lisa Roach.

itertools.islice now accepts integer-like objects (having an __index__ method) as start, stop, and slice arguments

Tokens needed for parsing in Python moved to C. `COMMENT`, `NL` and `ENCODING`. This way the tokens and tok_names in the token module don't get changed when you import the tokenize module.

Fixed parsing backslashes in f-strings.

Fixed various segfaults with dict when input collections are mutated during searching, inserting or comparing. Based on patches by Duane Griffin and Tim Mitchell.

Fixed type.__setattr__() and type.__delattr__() for non-interned attribute names. Based on patch by Eryk Sun.

If a KeyboardInterrupt happens when the interpreter is in the middle of resuming a chain of nested 'yield from' or 'await' calls, it's now correctly delivered to the innermost frame.

`object.__format__(x, '')` is now equivalent to `str(x)` rather than `format(str(self), '')`.

Circular imports involving absolute imports with binding a submodule to a name are now supported.

sys.getsizeof() on a code object now returns the sizes which includes the code struct and sizes of objects which it references. Patch by Dong-hee Na.

len() now raises ValueError rather than OverflowError if __len__() returned a large negative integer.

README.rst is now included in the list of distutils standard READMEs and therefore included in source distributions.

Fixed default implementations of __reduce__ and __reduce_ex__(). object.__reduce__() no longer takes arguments, object.__reduce_ex__() now requires one argument.

Fix memory usage regression of set and frozenset object.

Fixed error messages in the index() method of tuple, list and deque when pass indices of wrong type.

Shift operation now has less opportunity to raise OverflowError. ValueError always is raised rather than OverflowError for negative counts. Shifting zero with non-negative count always returns zero.

Fixed the slowing down to 25 times in the searching of some unlucky Unicode characters.

Add a unique ID to PyInterpreterState. This makes it easier to identify each subinterpreter.

The deprecation warning is emitted if __complex__ returns an instance of a strict subclass of complex. In a future versions of Python this can be an error.

Show correct error messages when any of the pthread_* calls in thread_pthread.h fails.

Fix a memory leak when an ImportError is raised during from import.

Fix an oversight that %b format for bytes should support objects follow the buffer protocol.

The `sys.path[0]` initialization change for bpo-29139 caused a regression by revealing an inconsistency in how sys.path is initialized when executing `__main__` from a zipfile, directory, or other import location. The interpreter now consistently avoids ever adding the import location's parent directory to `sys.path`, and ensures no other `sys.path` entries are inadvertently modified when inserting the import location named on the command line.

Escaped percent "%%" in the format string for classic string formatting no longer allows any characters between two percents.

Fix a regression that bytes format may fail when containing zero bytes inside.

bool(), float(), list() and tuple() no longer take keyword arguments. The first argument of int() can now be passes only as positional argument.

Set correct __cause__ for errors about invalid awaitables returned from __aiter__ and __anext__.

`bool(range)` works even if `len(range)` raises :exc:`OverflowError`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 717); *backlink*

Unknown interpreted text role "exc".

Fixes to memory allocation in _PyCode_SetExtra. Patch by Brian Coleman.

Fix minor regression of PyEval_CallObjectWithKeywords. It should raise TypeError when kwargs is not a dict. But it might cause segv when args=NULL and kwargs is not a dict.

Support __rmod__ for subclasses of str being called before str.__mod__. Patch by Martijn Pieters.

Fix stack_effect computation for CALL_FUNCTION_EX. Patch by Matthieu Dartiailh.

Fix incorrect handling of signed zeros in complex constructor for complex subclasses and for inputs having a __complex__ method. Patch by Serhiy Storchaka.

Fixed possibly dereferencing undefined pointers when creating weakref objects.

Add `docstring` field to Module, ClassDef, FunctionDef, and AsyncFunctionDef ast nodes. docstring is not first stmt in their body anymore. It affects `co_firstlineno` and `co_lnotab` of code object for module and class. (Reverted in :issue:`32911`.)

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 787); *backlink*

Unknown interpreted text role "issue".

Fixed use-after-free problem in key sharing dict.

Set the 'path' and 'name' attribute on ImportError for `from ... import ...`.

Improve from-import error message with location

If max_line_length=None is specified while using the Compat32 policy, it is no longer ignored. Patch by Mircea Cosbuc.

Prevent RunMainFromImporter overwriting sys.path[0].

Fixed possible BytesWarning when compare the code objects. Warnings could be emitted at compile time.

Fixed a crash when pass the iterable keyword argument to sorted().

Fix memory leak and use-after-free in os module (path_converter).

Fix regression in bytes(x) when x.__index__() raises Exception.

Call _PyObject_GC_TRACK() lazily when calling Python function. Calling function is up to 5% faster.

bytes.fromhex() and bytearray.fromhex() now ignore all ASCII whitespace, not only spaces. Patch by Robert Xiao.

Do not include <sys/random.h> if it does not exist.

Correct the positioning of the syntax error caret for indented blocks. Based on patch by Michael Layzell.

Fixed bytes formatting of octals with zero padding in alternate form.

Python function can now have more than 255 parameters. collections.namedtuple() now supports tuples with more than 255 elements.

The preferred encoding is UTF-8 on Android. Patch written by Chi Hsuan Yen.

Clean up interpreter startup (see PEP 432).

On Android, operating system data is now always encoded/decoded to/from UTF-8, instead of the locale encoding to avoid inconsistencies with os.fsencode() and os.fsdecode() which are already using UTF-8.

functools.lru_cache() was susceptible to an obscure reentrancy bug triggerable by a monkey-patched len() function.

Fix a memory leak in split-table dictionaries: setattr() must not convert combined table into split table. Patch written by INADA Naoki.

f-string expressions are no longer accepted as docstrings and by ast.literal_eval() even if they do not include expressions.

Fixed setting the offset attribute of SyntaxError by PyErr_SyntaxLocationEx() and PyErr_SyntaxLocationObject().

Fix the cross compilation of xxlimited when Python has been built with Py_DEBUG defined.

Rather than silently producing a class that doesn't support zero-argument `super()` in methods, failing to pass the new `__classcell__` namespace entry up to `type.__new__` now results in a `DeprecationWarning` and a class that supports zero-argument `super()`.

Modifying the class __dict__ inside the __set_name__ method of a descriptor that is used inside that class no longer prevents calling the __set_name__ method of other descriptors.

Remove the `PyEval_GetCallStats()` function and deprecate the untested and undocumented `sys.callstats()` function.

Remove the `CALL_PROFILE` special build: use the :func:`sys.setprofile` function, :mod:`cProfile` or :mod:`profile` to profile function calls.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 1043); *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]3.7.0a1.rst, line 1043); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 1043); *backlink*

Unknown interpreted text role "mod".

More than 255 arguments can now be passed to a function.

Fix a bug in the implementation `yield from` when checking if the next instruction is YIELD_FROM. Regression introduced by WORDCODE (issue #26647).

Fix error position of the unicode error in ASCII and Latin1 encoders when a string returned by the error handler contains multiple non-encodable characters (non-ASCII for the ASCII codec, characters out of the U+0000-U+00FF range for Latin1).

Optimize _PyDict_NewPresized() to create correct size dict. Improve speed of dict literal with constant keys up to 30%.

Show sys.version when -V option is supplied twice.

The with-statement now checks for __enter__ before it checks for __exit__. This gives less confusing error messages when both methods are missing. Patch by Jonathan Ellington.

Fix the set_inheritable() file descriptor method on platforms that do not have the ioctl FIOCLEX and FIONCLEX commands.

Fix not getting the locale's charset upon initializing the interpreter, on platforms that do not have langinfo.

Fixed crash in Py_DecodeLocale() in debug build on Mac OS X when decode astral characters. Patch by Xiang Zhang.

Improve speed of the STORE_DEREF opcode by 40%.

Extra slash no longer added to sys.path components in case of empty compile-time PYTHONPATH components.

Sped up converting int to float by reusing faster bits counting implementation. Patch by Adrian Wielgosik.

Optimize iterating split table values. Patch by Xiang Zhang.

PyDict_SetDefault didn't combine split table when needed. Patch by Xiang Zhang.

Deprecation warning for invalid str and byte escape sequences now prints better information about where the error occurs. Patch by Serhiy Storchaka and Eric Smith.

dict.update() no longer allocate unnecessary large memory.

Fixed potential crash in PyUnicode_AsDecodedObject() in debug build.

Fixed of-by-one error in the peephole optimizer that caused keeping unreachable code.

Improved exception reporting for problematic __set_name__ attributes.

Fixed possible memory leak in _PyTraceback_Add() and exception loss in PyTraceBack_Here().

Optimize and cleanup dict iteration.

Added C implementation of asyncio.Future. Original patch by Yury Selivanov.

Added sanity checks and tests for PyUnicode_CopyCharacters(). Patch by Xiang Zhang.

The type of long range iterator is now registered as Iterator. Patch by Oren Milman.

Creating instances of range_iterator by calling range_iterator type now is disallowed. Calling iter() on range instance is the only way. Patch by Oren Milman.

Resolving special methods of uninitialized type now causes implicit initialization of the type instead of a fail.

PyType_Ready() now checks that tp_name is not NULL. Original patch by Niklas Koep.

Fixed possible crash when AST is changed in process of compiling it.

Dict reduces possibility of 2nd conflict in hash table when hashes have same lower bits.

String constants with null character no longer interned.

Fix crash when GC runs during weakref callbacks.

String constants now interned recursively in tuples and frozensets.

ImportError.__init__ now resets not specified attributes.

Fixed misleading error message when ImportError called with invalid keyword args.

Fix incorrect type in complex(1.0, {2:3}) error message. Patch by Soumya Sharma.

Single var-positional argument of tuple subtype was passed unscathed to the C-defined function. Now it is converted to exact tuple.

Now __set_name__ is looked up on the class instead of the instance.

Fallback on reading /dev/urandom device when the getrandom() syscall fails with EPERM, for example when blocked by SECCOMP.

Don't import readline in isolated mode.

Remove some redundant assignments to ob_size in longobject.c. Thanks Oren Milman.

Clean up redundant code in long_rshift function. Thanks Oren Milman.

Upgrade internal unicode databases to Unicode version 9.0.0.

Fix a regression in zipimport's compile_source(). zipimport should use the same optimization level as the interpreter.

Replace Py_MEMCPY with memcpy(). Visual Studio can properly optimize memcpy().

Fix dict.pop() for splitted dictionary when trying to remove a "pending key" (Not yet inserted in split-table). Patch by Xiang Zhang.

Raise DeprecationWarning when async and await keywords are used as variable/attribute/class/function name.

Fix a refleak in code that raises DeprecationWarning.

Fix asynchronous generators aclose() and athrow() to handle StopAsyncIteration propagation properly.

Speed-up method calls: add LOAD_METHOD and CALL_METHOD opcodes.

xml.etree: Fix a crash when a parser is part of a reference cycle.

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

typing.get_type_hints now finds the right globalns for classes and modules by default (when no `globalns` was specified by the caller).

Speed improvements to the `typing` module. Original PRs by Ivan Levkivskyi and Mitar.

The C accelerator module of ElementTree ignored exceptions raised when looking up TreeBuilder target methods in XMLParser().

socket.create_connection() now fixes manually a reference cycle: clear the variable storing the last exception on success.

LoggerAdapter objects can now be nested.

SSLContext.check_hostname now automatically sets SSLContext.verify_mode to ssl.CERT_REQUIRED instead of failing with a ValueError.

socketserver.ThreadingMixIn now keeps a list of non-daemonic threads to wait until all these threads complete in server_close().

Changed the implementation strategy for collections.namedtuple() to substantially reduce the use of exec() in favor of precomputed methods. As a result, the *verbose* parameter and *_source* attribute are no longer supported. The benefits include 1) having a smaller memory footprint for applications using multiple named tuples, 2) faster creation of the named tuple class (approx 4x to 6x depending on how it is measured), and 3) minor speed-ups for instance creation using __new__, _make, and _replace. (The primary patch contributor is Jelle Zijlstra with further improvements by INADA Naoki, Serhiy Storchaka, and Raymond Hettinger.)

Improves SSL error handling to avoid losing error numbers.

Make return types of SSLContext.wrap_bio() and SSLContext.wrap_socket() customizable.

ssl.SSLContext() now uses OpenSSL error information when a context cannot be instantiated.

The SSL module now raises SSLCertVerificationError when OpenSSL fails to verify the peer's certificate. The exception contains more information about the error.

SSLSocket.sendall() now uses memoryview to create slices of data. This fixes support for all bytes-like object. It is also more efficient and avoids costly copies.

A new function `argparse.ArgumentParser.parse_intermixed_args` provides the ability to parse command lines where there user intermixes options and positional arguments.

Fix string concatenation bug in rare error path in the subprocess module

Micro-optimize :func:`asyncio._get_running_loop` to become up to 10% faster.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst`, line 1705); *backlink*

Unknown interpreted text role "func".

expat: Update libexpat from 2.2.3 to 2.2.4. Fix copying of partial characters for UTF-8 input (libexpat bug 115): https://github.com/libexpat/libexpat/issues/115

Add TLS 1.3 cipher suites and OP_NO_TLSv1_3.

`string.Template` subclasses can optionally define `braceidpattern` if they want to specify different placeholder patterns inside and outside the braces. If None (the default) it falls back to `idpattern`.

concurrent.futures.ProcessPoolExecutor.shutdown() now explicitly closes the call queue. Moreover, shutdown(wait=True) now also join the call queue thread, to prevent leaking a dangling thread.

The `map()` and `as_completed()` iterators in `concurrent.futures` now avoid keeping a reference to yielded objects.

Fix `fileinput.FileInput(files, inplace=True)` when `files` contain `pathlib.Path` objects.

Fix ctypes producing wrong PEP 3118 type codes for integer types.

`AF_VSOCK` has been added to the socket interface which allows communication between virtual machines and their host.

The subprocess module now sets the filename when FileNotFoundError is raised on POSIX systems due to the executable or cwd not being found.

Update some methods in the _pyio module to also accept integer types. Patch by Oren Milman.

concurrent.futures: WorkItem.run() used by ThreadPoolExecutor now breaks a reference cycle between an exception object and the WorkItem object.

xmlrpc.server now explicitly breaks reference cycles when using sys.exc_info() in code handling exceptions.

configparser: reading defaults in the `ConfigParser()` constructor is now using `read_dict()`, making its behavior consistent with the rest of the parser. Non-string keys and values in the defaults dictionary are now being implicitly converted to strings. Patch by James Tocknell.

pydoc: the stop() method of the private ServerThread class now waits until DocServer.serve_until_quit() completes and then explicitly sets its docserver attribute to None to break a reference cycle.

Many asserts in *multiprocessing* are now more informative, and some error types have been changed to more specific ones.

Convert zipimport to use Argument Clinic.

The ssl and hashlib modules now call OPENSSL_add_all_algorithms_noconf() on OpenSSL < 1.1.0. The function detects CPU features and enables optimizations on some CPU architectures such as POWER8. Patch is based on research from Gustavo Serra Scalet.

Non-daemonic threads created by a multiprocessing.Process are now joined on child exit.

dis now works with asynchronous generator and coroutine objects. Patch by George Collins based on diagnosis by Luciano Ramalho.

There are a number of uninformative asserts in the *multiprocessing* module, as noted in issue 5001. This change fixes two of the most potentially problematic ones, since they are in error-reporting code, in the *multiprocessing.managers.convert_to_error* function. (It also makes more informative a ValueError message.) The only potentially problematic change is that the AssertionError is now a TypeError; however, this should also help distinguish it from an AssertionError being *reported* by the function/its caller (such as in issue 31169). - Patch by Allen W. Smith (drallensmith on github).

Fixed miscellaneous errors in asyncio speedup module.

socketserver.ForkingMixIn.server_close() now waits until all child processes completed to prevent leaking zombie processes.

Add an `include_file` parameter to `zipapp.create_archive()`

Optimize array.array comparison. It is now from 10x up to 70x faster when comparing arrays holding values of the same integer type.

ttk: fix the destroy() method of LabeledScale and OptionMenu classes. Call the parent destroy() method even if the used attribute doesn't exist. The LabeledScale.destroy() method now also explicitly clears label and scale attributes to help the garbage collector to destroy all widgets.

Fix *copyreg._slotnames()* mangled attribute calculation for classes whose name begins with an underscore. Patch by Shane Harvey.

Allow *logging.config.fileConfig* to accept kwargs and/or args.

`pathlib.Path` objects now include an `is_mount()` method (only implemented on POSIX). This is similar to `os.path.ismount(p)`. Patch by Cooper Ry Lees.

Fixed a crash when using asyncio and threads.

Added support for CAN ISO-TP protocol in the socket module.

Added a `setStream` method to `logging.StreamHandler` to allow the stream to be set after creation.

Fix handling of long oids in ssl. Based on patch by Christian Heimes.

Support tzinfo objects with sub-minute offsets.

Fix shared memory performance regression in multiprocessing in 3.x. Shared memory used anonymous memory mappings in 2.x, while 3.x mmaps actual files. Try to be careful to do as little disk I/O as possible.

Fix too many fds in processes started with the "forkserver" method. A child process would inherit as many fds as the number of still-running children.

Fix `unittest.mock`'s autospec to not fail on method-bound builtin functions. Patch by Aaron Gallagher.

Fix decrementing a borrowed reference in tracemalloc.

Fix multiprocessing.sharedctypes to recognize typecodes `'q'` and `'Q'`.

Remove obsolete code in readline module for platforms where GNU readline is older than 2.1 or where select() is not available.

Change `ttk.OptionMenu` radiobuttons to be unique across instances of `OptionMenu`.

Fix multiprocessing.Queue.join_thread(): it now waits until the thread completes, even if the thread was started by the same process which created the queue.

Fix segfault in readline when using readline's history-size option. Patch by Nir Soffer.

Added multiprocessing.Process.kill method to terminate using the SIGKILL signal on Unix.

socket.close() now ignores ECONNRESET error.

Fix out of bounds write in *asyncio.CFuture.remove_done_callback()*.

Use keywords in the `repr` of `datetime.timedelta`.

signal.setitimer() may disable the timer when passed a tiny value. Tiny values (such as 1e-6) are valid non-zero values for setitimer(), which is specified as taking microsecond-resolution intervals. However, on some platform, our conversion routine could convert 1e-6 into a zero interval, therefore disabling the timer instead of (re-)scheduling it.

Fix bug when modifying os.environ while iterating over it

Avoid importing `sysconfig` from `site` to improve startup speed. Python startup is about 5% faster on Linux and 30% faster on macOS.

Add missing parameter "n" on multiprocessing.Condition.notify(). The doc claims multiprocessing.Condition behaves like threading.Condition, but its notify() method lacked the optional "n" argument (to specify the number of sleepers to wake up) that threading.Condition.notify() accepts.

Fix email header value parser dropping folding white space in certain cases.

Add a `close()` method to `multiprocessing.Process`.

Fix a segmentation fault in _hashopenssl when standard hash functions such as md5 are not available in the linked OpenSSL library. As in some special FIPS-140 build environments.

Update zlib to 1.2.11.

ftplib.FTP.putline() now throws ValueError on commands that contains CR or LF. Patch by Dong-hee Na.

os.listdir() and os.scandir() now emit bytes names when called with bytes-like argument.

Prohibited the '=' character in environment variable names in `os.putenv()` and `os.spawn*()`.

The description of a unittest subtest now preserves the order of keyword arguments of TestCase.subTest().

struct.Struct.format type is now :class:`str` instead of :class:`bytes`.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 2302); *backlink***
>
> Unknown interpreted text role "class".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 2302); *backlink***
>
> Unknown interpreted text role "class".

Fix concurrent.futures.thread.ThreadPoolExecutor threads to have a non repr() based thread name by default when no thread_name_prefix is supplied. They will now identify themselves as "ThreadPoolExecutor-y_n".

Fixed the lgettext() family of functions in the gettext module. They now always return bytes.

Functional API of enum allows to create empty enums. Patched by Dong-hee Na

Fix race condition between signal delivery and wakeup file descriptor. Patch by Nathaniel Smith.

lib2to3 now recognizes `rb'...'` and `f'...'` strings.

pkgutil.walk_packages function now raises ValueError if *path* is a string. Patch by Sanyam Khurana.

Avoid race condition in multiprocessing cleanup.

Fix multiprocessing.Process.exitcode to return the opposite of the signal number when the process is killed by a signal (instead of 255) when using the "forkserver" method.

The traceback no longer displayed for SystemExit raised in a callback registered by atexit.

Don't log exceptions if Task/Future "cancel()" method was called.

Fix path calculation in *imp.load_package()*, fixing it for cases when a package is only shipped with bytecodes. Patch by Alexandru Ardelean.

The dis.dis() function now is able to disassemble nested code objects.

selectors does not take KeyboardInterrupt and SystemExit into account, leaving a fd in a bad state in case of error. Patch by Giampaolo Rodola'.

multiprocessing.Queue.get() with a timeout now polls its reader in non-blocking mode if it succeeded to acquire the lock but the acquire took longer than the timeout.

Updates to typing module: Add generic AsyncContextManager, add support for ContextManager on all versions. Original PRs by Jelle Zijlstra and Ivan Levkivskyi

re.compile() no longer raises a BytesWarning when compiling a bytes instance with misplaced inline modifier. Patch by Roy Williams.

Fix ssl sockets leaks when connection is aborted in asyncio/ssl implementation. Patch by Michaël Sghaïer.

Closing transport during handshake process leaks open socket. Patch by Nikolay Kim

Fix waiter cancellation in asyncio.Lock. Patch by Mathieu Sornay.

modify() method of poll(), epoll() and devpoll() based classes of selectors module is around 10% faster. Patch by Giampaolo Rodola'.

On Windows, subprocess.Popen.communicate() now also ignore EINVAL on stdin.write() if the child process is still running but closed the pipe.

Addded empty __slots__ to abc.ABC. This allows subclassers to deny __dict__ and __weakref__ creation. Patch by Aaron Hall.

Loggers are now pickleable.

faulthandler now correctly filters and displays exception codes on Windows

Add TextIOWrapper.reconfigure() and a TextIOWrapper.write_through attribute.

Fix possible overflow when organize struct.pack_into error message. Patch by Yuan Liu.

Fix the problem that logging.handlers.SysLogHandler cannot handle IPv6 addresses.

Allow registering at-fork handlers.

Deprecate invalid ctypes call protection on Windows. Patch by Mariatta Wijaya.

multiprocessing.Queue._feed background running thread do not break from main loop on exception.

Fix handling escape characters in HZ codec. Based on patch by Ma Lin.

inspect.signature() now supports callables with variable-argument parameters wrapped with partialmethod. Patch by Dong-hee Na.

importlib.find_spec() raises ModuleNotFoundError instead of AttributeError if the specified parent module is not a package (i.e. lacks a __path__ attribute).

Fix AttributeError when using SimpleQueue.empty() under *spawn* and *forkserver* start methods.

Warnings emitted when compile a regular expression now always point to the line in the user code. Previously they could point into inners of the re module if emitted from inside of groups or conditionals.

imaplib and poplib now catch the Windows socket WSAEINVAL error (code 10022) on shutdown(SHUT_RDWR): An invalid operation was attempted. This error occurs sometimes on SSL connections.

Removed previously deprecated in Python 2.4 classes Plist, Dict and _InternalDict in the plistlib module. Dict values in the result of functions readPlist() and readPlistFromBytes() are now normal dicts. You no longer can use attribute access to access items of these dictionaries.

The :mod:`macpath` is now deprecated and will be removed in Python 3.8.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst`, **line 2679**); *backlink*
>
> Unknown interpreted text role "mod".

Compiling regular expression in debug mode on CPython now displays the compiled bytecode in human readable form.

Fixed `Task.cancel()` can be ignored when the task is running coroutine and the coroutine returned without any more `await`.

contextlib.AbstractContextManager now supports anti-registration by setting __enter__ = None or __exit__ = None, following the pattern introduced in bpo-25958. Patch by Jelle Zijlstra.

Enhanced regular expressions optimization. This increased the performance of matching some patterns up to 25 times.

Weaken the condition of deprecation warnings for inline modifiers. Now allowed several subsequential inline modifiers at the start of the pattern (e.g. `'(?i)(?s)...'`). In verbose mode whitespaces and comments now are allowed before and between inline modifiers (e.g. `'(?x) (?i) (?s)...'`).

Optimized case-insensitive matching and searching of regular expressions.

Fix range checking in GB18030 decoder. Original patch by Ma Lin.

rewrite cgi.parse_multipart, reusing the FieldStorage class and making its results consistent with those of FieldStorage for multipart/form-data requests. Patch by Pierre Quentel.

Removed the __init__ methods of _json's scanner and encoder. Misusing them could cause memory leaks or crashes. Now scanner and encoder objects are completely initialized in the __new__ methods.

Compiled regular expression objects with the re.LOCALE flag no longer depend on the locale at compile time. Only the locale at matching time affects the result of matching.

Avoid KeyboardInterrupt tracebacks in forkserver helper process when Ctrl-C is received.

binascii.b2a_uu() and uu.encode() now support using `` ` `` as zero instead of space.

Various updates to typing module: add typing.NoReturn type, use WrapperDescriptorType, minor bug-fixes. Original PRs by Jim Fasarakis-Hilliard and Ivan Levkivskyi.

Fix getsockname() for unbound AF_UNIX sockets on Linux.

The seek() and tell() methods of io.FileIO now set the internal seekable attribute to avoid one syscall on open() (in buffered or text mode).

unittest's assertAlmostEqual and assertNotAlmostEqual provide a better message in case of failure which includes the difference between left and right arguments. (patch by Giampaolo Rodola')

Add support for curses.A_ITALIC.

inspect.isabstract() now works during __init_subclass__. Patch by Nate Soares.

Preserve generator state when _random.Random.setstate() raises an exception. Patch by Bryan Olson.

Fixed leaks and crashes in errors handling in the parser module.

Column widths in the output of dis.dis() are now adjusted for large line numbers and instruction offsets.

Fixed crashes in IOBase methods __next__() and readlines() when readline() or __next__() respectively return non-sizeable object. Fixed possible other errors caused by not checking results of PyObject_Size(), PySequence_Size(), or PyMapping_Size().

Fix PathLike support for shutil.unpack_archive. Patch by Jelle Zijlstra.

Compiled regular expression and match objects in the re module now support copy.copy() and copy.deepcopy() (they are considered atomic).

_io._IOBase.readlines will check if it's closed first when hint is present.

Fixed race condition in pathlib mkdir with flags parents=True. Patch by Armin Rigo.

Fixed arbitrary unchaining of RuntimeError exceptions in contextlib.contextmanager. Patch by Siddharth Velankar.

Test that sqlite3 trace callback is not called multiple times when schema is changing. Indirectly fixed by switching to use sqlite3_prepare_v2() in bpo-9303. Patch by Aviv Palivoda.

Allowed calling the close() method of the zip entry writer object multiple times. Writing to a closed writer now always produces a ValueError.

Pickling and copying ImportError now preserves name and path attributes.

re.escape() now escapes only regex special characters.

Add math.remainder operation, implementing remainder as specified in IEEE 754.

Improve struct.pack_into() exception messages for problems with the buffer size and offset. Patch by Andrew Nester.

Support If-Modified-Since HTTP header (browser cache). Patch by Pierre Quentel.

Fixed comparison check for ipaddress.ip_interface objects. Patch by Sanjay Sundaresan.

Fixed memory leaks in the replace() method of datetime and time objects when pass out of bound fold argument.

Fix a crash in itertools.chain.from_iterable when encountering long runs of empty iterables.

Sped up reading encrypted ZIP files by 2 times.

Element.getiterator() and the html parameter of XMLParser() were deprecated only in the documentation (since Python 3.2 and 3.4 correspondingly). Now using them emits a deprecation warning.

Fixed multiple crashes in ElementTree caused by race conditions and wrong types.

Added support of file descriptors in os.scandir() on Unix. os.fwalk() is sped up by 2 times by using os.scandir().

Fixed a bug in pools in multiprocessing.pool that raising an exception at the very first of an iterable may swallow the exception or make the program hang. Patch by Davin Potts and Xiang Zhang.

unittest.TestCase.assertRaises() now manually breaks a reference cycle to not keep objects alive longer than expected.

The zipapp module now supports general path-like objects, not just pathlib.Path.

Avoid incorrect errors raised by Path.mkdir(exist_ok=True) when the OS gives priority to errors such as EACCES over EEXIST.

Release references to tasks, their arguments and their results as soon as they are finished in multiprocessing.Pool.

The mode argument of os.makedirs() no longer affects the file permission bits of newly-created intermediate-level directories.

faulthandler: Restore the old sigaltstack during teardown. Patch by Christophe Zeitouny.

Fixed crashes in repr of recursive buffered file-like objects.

Fix crashes in partial.__repr__ if the keys of partial.keywords are not strings. Patch by Michael Seifert.

Fixed possible failing or crashing input() if attributes "encoding" or "errors" of sys.stdin or sys.stdout are not set or are not strings.

Using non-integer value for selecting a plural form in gettext is now deprecated.

Use C library implementation for math functions erf() and erfc().

os.stat() and os.DirEntry.inode() now convert inode (st_ino) using unsigned integers.

Fix a bug that prevented array 'Q', 'L' and 'I' from accepting big intables (objects that have __int__) as elements.

Speed up importing the webbrowser module. webbrowser.register() is now thread-safe.

The zipfile module now accepts path-like objects for external paths.

index() and count() methods of collections.abc.Sequence now check identity before checking equality when do comparisons.

Added support for bytes paths in os.fwalk().

Add new :data:`socket.TCP_NOTSENT_LOWAT` (Linux 3.12) constant. Patch by Nathaniel J. Smith.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst`, line 3277); *backlink***
>
> Unknown interpreted text role "data".

Allow use of path-like object as a single argument in ConfigParser.read(). Patch by David Ellis.

Migrate sqlite3 module to _v2 API. Patch by Aviv Palivoda.

Fix out of bound iteration in asyncio.Future.remove_done_callback implemented in C.

asyncio.subprocess.SubprocessStreamProtocol no longer closes before all pipes are closed.

Fix Task.current_task and Task.all_tasks implemented in C to accept None argument as their pure Python implementation.

Fix asyncio to support instantiation of new event loops in child processes.

SimpleXMLRPCDispatcher no longer chains KeyError (or any other exception) to exception(s) raised in the dispatched methods. Patch by Petr Motejlek.

Method register_function() of xmlrpc.server.SimpleXMLRPCDispatcher and its subclasses can now be used as a decorator.

Fix assertion error in threading._DummyThread.is_alive().

Add a test that checks that cwd parameter of Popen() accepts PathLike objects. Patch by Sayan Chowdhury.

Start a transaction implicitly before a DML statement. Patch by Aviv Palivoda.

get_extra_info() raises exception if get called on closed ssl transport. Patch by Nikolay Kim.

urllib.parse.quote is now based on RFC 3986 and hence includes '~' in the set of characters that is not quoted by default. Patch by Christian Theune and Ratnadeep Debnath.

Altering a kwarg dictionary passed to functools.partial() no longer affects a partial object after creation.

Fix file object leak in aifc.open() when file is given as a filesystem path and is not in valid AIFF format. Patch by Anthony Zhang.

Add uuid.SafeUUID and uuid.UUID.is_safe to relay information from the platform about whether generated UUIDs are generated with a multiprocessing safe method.

Improve some deprecations in importlib. Some deprecated methods now emit DeprecationWarnings and have better descriptive

messages.

Fixed different behaviour of Decimal.from_float() for _decimal and _pydecimal. Thanks Andrew Nester.

locale.format_string now supports the 'monetary' keyword argument, and locale.format is deprecated.

importlib.reload() now raises ModuleNotFoundError if the module lacks a spec.

Various updates to typing module: typing.Counter, typing.ChainMap, improved ABC caching, etc. Original PRs by Jelle Zijlstra, Ivan Levkivskyi, Manuel Krebber, and Łukasz Langa.

Fix datetime.fromtimestamp() regression introduced in Python 3.6.0: check minimum and maximum years.

Prevent infinite loop in pathlib.Path.mkdir

Fixed out-of-bounds buffer access in the group() method of the match object. Based on patch by WGH.

Add WrapperDescriptorType, MethodWrapperType, and MethodDescriptorType built-in types to types module. Original patch by Manuel Krebber.

Unused install_misc command is now removed. It has been documented as unused since 2000. Patch by Eric N. Vander Weele.

The extend() method is now called instead of the append() method when unpickle collections.deque and other list-like objects. This can speed up unpickling to 2 times.

The help of a builtin or extension class now includes the constructor signature if __text_signature__ is provided for the class.

Fix subprocess.Popen.wait() when the child process has exited to a stopped instead of terminated state (ex: when under ptrace).

Fix a regression in argparse that help messages would wrap at non-breaking spaces.

Fixed the comparison of mock.MagickMock with mock.ANY.

Removed deprecated function ntpath.splitunc().

Removed support of deprecated argument "exclude" in tarfile.TarFile.add().

Fixed infinite recursion in the repr of uninitialized ctypes.CDLL instances.

Removed deprecated features in the http.cookies module.

A format string argument for string.Formatter.format() is now positional-only.

Removed support of deprecated undocumented keyword arguments in methods of regular expression objects.

Fixed race condition in C implementation of functools.lru_cache. KeyError could be raised when cached function with full cache was simultaneously called from different threads with the same uncached arguments.

The unittest.mock.sentinel attributes now preserve their identity when they are copied or pickled.

In urllib.request, suffixes in no_proxy environment variable with leading dots could match related hostnames again (e.g. .b.c matches a.b.c). Patch by Milan Oberkirch.

Fix unittest.mock._Call helper: don't ignore the name parameter anymore. Patch written by Jiajun Huang.

inspect.getframeinfo() now correctly shows the first line of a context. Patch by Sam Breese.

Update authorizer constants in sqlite3 module. Patch by Dingyuan Wang.

Prevent infinite loop in pathlib.resolve() on Windows

Fixed recursion errors in large or resized curses.textpad.Textbox. Based on patch by Tycho Andersen.

curses.ascii predicates now work correctly with negative integers.

old keys should not remove new values from WeakValueDictionary when collecting from another thread.

Remove editor artifacts from Tix.py.

Fixed a crash when deallocate deep ElementTree.

Fix bugs in WeakValueDictionary.setdefault() and WeakValueDictionary.pop() when a GC collection happens in another thread.

Fixed a crash in resource.prlimit() when passing a sequence that doesn't own its elements as limits.

subprocess.Popen uses /system/bin/sh on Android as the shell, instead of /bin/sh.

multiprocessing.set_forkserver_preload() would crash the forkserver process if a preloaded module instantiated some multiprocessing objects such as locks.

The chown() method of the tarfile.TarFile class does not fail now when the grp module cannot be imported, as for example on Android platforms.

dbm.dumb now supports reading read-only files and no longer writes the index file when it is not changed. A deprecation warning is now emitted if the index file is missed and recreated in the 'r' and 'w' modes (will be an error in future Python releases).

Unknown escapes consisting of `'\'` and an ASCII letter in re.sub() replacement templates regular expressions now are errors.

Fix a regression introduced in warnings.catch_warnings(): call warnings.showwarning() if it was overridden inside the context

manager.

To assist with upgrades from 2.7, the previously documented deprecation of `inspect.getfullargspec()` has been reversed. This decision may be revisited again after the Python 2.7 branch is no longer officially supported.

Add sys.getandroidapilevel(): return the build time API version of Android as an integer. Function only available on Android.

Add new :data:`socket.TCP_CONGESTION` (Linux 2.6.13) and :data:`socket.TCP_USER_TIMEOUT` (Linux 2.6.37) constants. Patch written by Omar Sandoval.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst`, **line 3874**); *backlink*
>
> Unknown interpreted text role "data".

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst`, **line 3874**); *backlink*
>
> Unknown interpreted text role "data".

Restored the __reduce__() methods of datetime objects.

Regular expression patterns, _sre.SRE_Pattern objects created by re.compile(), become comparable (only x==y and x!=y operators). This change should fix the issue #18383: don't duplicate warning filters when the warnings module is reloaded (thing usually only done in unit tests).

Remove the subprocess.Popen.wait endtime parameter. It was deprecated in 3.4 and undocumented prior to that.

In ctypes, prevent a crash calling the from_buffer() and from_buffer_copy() methods on abstract classes like Array.

In the "http.server" module, parse the protocol version if possible, to avoid using HTTP 0.9 in some error responses.

Makes Path.resolve() succeed on paths that do not exist. Patch by Vajrasky Kok

Fixed possible DoS and arbitrary code execution when handle plural form selections in the gettext module. The expression parser now supports exact syntax supported by GNU gettext.

Fixed possible crash in _io.TextIOWrapper deallocator when the garbage collector is invoked in other thread. Based on patch by Sebastian Cufre.

LZMA compressor and decompressor no longer raise exceptions if given empty data twice. Patch by Benjamin Fogle.

Fixed segfault in curses's addch() with ncurses6.

tarfile.open() with mode "r" or "r:" now tries to open a tar file with compression before trying to open it without compression. Otherwise it had 50% chance failed with ignore_zeros=True.

The webbrowser module now supports Firefox 36+ and derived browsers. Based on patch by Oleg Broytman.

The webbrowser in an X environment now prefers using the default browser directly. Also, the webbrowser register() function now has a documented 'preferred' argument, to specify browsers to be returned by get() with no arguments. Patch by David Steele

Fixed bugs in tkinter.ttk.LabeledScale and tkinter.Scale caused by representing the scale as float value internally in Tk. tkinter.IntVar now works if float value is set to underlying Tk variable.

calendar.TextCalendar.prweek() no longer prints a space after a weeks's calendar. calendar.TextCalendar.pryear() no longer prints redundant newline after a year's calendar. Based on patch by Xiang Zhang.

calendar.TextCalendar.prmonth() no longer prints a space at the start of new line after printing a month's calendar. Patch by Xiang Zhang.

The textwrap.TextWrapper class now honors non-breaking spaces. Based on patch by Kaarle Ritvanen.

os.fwalk() no longer fails on broken links.

Fix iterator of C implemented asyncio.Future doesn't accept non-None value is passed to it.send(val).

Generated names for Tkinter widgets now start by the "!" prefix for readability.

Fixed HList.header_exists() in tkinter.tix module by addin a workaround to Tix library bug.

shutil.make_archive() no longer adds entry "./" to ZIP archive.

re.sub() now raises an error for invalid numerical group reference in replacement template even if the pattern is not found in the string. Error message for invalid group reference now includes the group index and the position of the reference. Based on patch by SilentGhost.

timeit now uses the sequence 1, 2, 5, 10, 20, 50,... instead of 1, 10, 100,... for autoranging.

Command-line interface of the zipfile module now uses argparse. Added support of long options.

Optimize csv.DictWriter for large number of columns. Patch by Mariatta Wijaya.

Fix C implemented asyncio.Future didn't work on Windows.

In the "io" module, the argument to BufferedReader and BytesIO's read1() methods is now optional and can be -1, matching the BufferedIOBase specification.

Fix error building socket module when multithreading is disabled.

timeit: remove `-c`/`--clock` and `-t`/`--time` command line options which were deprecated since Python 3.3.

timeit now repeats the benchmarks 5 times instead of only 3 to make benchmarks more reliable.

timeit autorange now uses a single loop iteration if the benchmark takes less than 10 seconds, instead of 10 iterations. "python3 -m timeit -s 'import time' 'time.sleep(1)'" now takes 4 seconds instead of 40 seconds.

Distutils.sdist now looks for README and setup.py files with case sensitivity. This behavior matches that found in Setuptools 6.0 and later. See setuptools 100 for rationale.

Make webbrowser support Chrome on Mac OS X. Patch by Ned Batchelder.

Fix references leaked by pdb in the handling of SIGINT handlers.

Fixed bytes path support in os.scandir() on Windows. Patch by Eryk Sun.

The disassembler now decodes FORMAT_VALUE argument.

unittest.mock Mock autospec functions now properly support assert_called, assert_not_called, and assert_called_once.

lzma module now supports pathlib.

Fixed writing non-BMP characters with binary format in plistlib.

bz2 module now supports pathlib. Initial patch by Ethan Furman.

gzip now supports pathlib. Patch by Ethan Furman.

Deprecated silent truncations in socket.htons and socket.ntohs. Original patch by Oren Milman.

Optimized merging var-keyword arguments and improved error message when passing a non-mapping as a var-keyword argument.

Improved error message when passing a non-iterable as a var-positional argument. Added opcode BUILD_TUPLE_UNPACK_WITH_CALL.

Fixed possible crashes when unpickle itertools objects from incorrect pickle data. Based on patch by John Leitch.

imghdr now supports pathlib.

compileall now supports pathlib.

Fix function declaration (C flags) for the getiterator() method of xml.etree.ElementTree.Element.

Stop using localtime() and gmtime() in the time module. Introduced platform independent _PyTime_localtime API that is similar to POSIX localtime_r, but available on all platforms. Patch by Ed Schouten.

Fixed calendar functions for extreme months: 0001-01 and 9999-12. Methods itermonthdays() and itermonthdays2() are reimplemented so that they don't call itermonthdates() which can cause datetime.date under/overflow.

Fixed possible use after free in the decompress() methods of the LZMADecompressor and BZ2Decompressor classes. Original patch by John Leitch.

Fixed possible crash in sqlite3.Connection.create_collation() if pass invalid string-like object as a name. Patch by Xiang Zhang.

random.choices() now has k as a keyword-only argument to improve the readability of common cases and come into line with the signature used in other languages.

Fix invalid exception handling in Lib/ctypes/macholib/dyld.py. Patch by Madison May.

Fixed support of default root window in the tkinter.tix module. Added the master parameter in the DisplayStyle constructor.

In the traceback module, restore the formatting of exception messages like "Exception: None". This fixes a regression introduced in 3.5a2.

Allow falsy values to be used for msg parameter of subTest().

Fix a memory leak in os.getrandom() when the getrandom() is interrupted by a signal and a signal handler raises a Python exception.

Fix memory leak on Windows in the os module (fix path_converter() function).

RobotFileParser now correctly returns default values for crawl_delay and request_rate. Initial patch by Peter Wirtz.

Prevent memory leak in win32_ver().

Fix UnboundLocalError in socket._sendfile_use_sendfile.

Check for ERROR_ACCESS_DENIED in Windows implementation of os.stat(). Patch by Eryk Sun.

Warning message emitted by using inline flags in the middle of regular expression now contains a (truncated) regex pattern. Patch by Tim Graham.

Prevent codecs.escape_encode() from raising SystemError when an empty bytestring is passed.

Get antigravity over HTTPS. Patch by Kaartic Sivaraam.

Enable WebSocket URL schemes in urllib.parse.urljoin. Patch by Gergely Imreh and Markus Holtermann.

Fix a crash in parse_envlist() when env contains byte strings. Patch by Eryk Sun.

Fixed buffer overrun in binascii.b2a_qp() and binascii.a2b_qp().

Fix socket accept exhaustion during high TCP traffic. Patch by Kevin Conway.

Handle when SO_REUSEPORT isn't properly supported. Patch by Seth Michael Larson.

Inspect functools.partial in asyncio.Handle.__repr__. Patch by iceboy.

Fix slow pipes IO in asyncio. Patch by INADA Naoki.

Fix callbacks race in asyncio.SelectorLoop.sock_connect.

Fix selectors incorrectly retain invalid file descriptors. Patch by Mark Williams.

Remove vestigial MacOS 9 macurl2path module and its tests.

Refuse monitoring processes if the child watcher has no loop attached. Patch by Vincent Michel.

Raise RuntimeError when transport's FD is used with add_reader, add_writer, etc.

Speedup asyncio.StreamReader.readexactly. Patch by Коренберг Марк.

Deprecate passing asyncio.Handles to run_in_executor.

Fix asyncio to support formatting of non-python coroutines.

Remove UNIX socket from FS before binding. Patch by Коренберг Марк.

Prohibit Tasks to await on themselves.

Reading a corrupt config file left configparser in an invalid state. Original patch by Florian Höch.

ABCMeta.__new__ now accepts `**kwargs`, allowing abstract base classes to use keyword parameters in __init_subclass__. Patch by Nate Soares.

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.

path.resolve(strict=False) no longer cuts the path after the first element not present in the filesystem. Patch by Antoine Pietri.

Fix incomplete code snippet in the ZeroMQSocketListener and ZeroMQSocketHandler examples and adapt them to Python 3.

Add RFC 7525 and Mozilla server side TLS links to SSL documentation.

Allow the pydoc server to bind to arbitrary hostnames.

Clarify doc on truth value testing. Original patch by Peter Thomassen.

Add missing attribute related constants in curses documentation.

the link targets for :func:`bytes` and :func:`bytearray` are now their respective type definitions, rather than the corresponding builtin function entries. Use :ref:`bytes <func-bytes>` and :ref:`bytearray <func-bytearray>` to reference the latter. In order to ensure this and future cross-reference updates are applied automatically, the daily documentation builds now disable the default output caching features in Sphinx.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 4797); *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]3.7.0a1.rst, line 4797); *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]3.7.0a1.rst, line 4797); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\[cpython-main][Misc][NEWS.d]3.7.0a1.rst, line 4797); *backlink*

Unknown interpreted text role "ref".

Add missing info of code object in inspect documentation.

Improve the documentation for, and links to, template strings by emphasizing their utility for internationalization, and by clarifying some usage constraints. (See also: bpo-20314, bpo-12518)

Link the documentation to its source file on GitHub.

Document smtpd.py as effectively deprecated and add a pointer to aiosmtpd, a third-party asyncio-based replacement.

Add canonical header link on each page to corresponding major version of the documentation. Patch by Matthias Bussonnier.

Fix Python 2 syntax in code for building the documentation.

The data model reference and the porting section in the 3.6 What's New guide now cover the additional `__classcell__` handling needed for custom metaclasses to fully support PEP 487 and zero-argument `super()`.

Documented command-line interface of zipfile.

test.support.HOST is now "localhost", a new HOSTv4 constant has been added for your `127.0.0.1` needs, similar to the existing HOSTv6 constant.

Silence traceback in test_ssl

Prefer PROTOCOL_TLS_CLIENT and PROTOCOL_TLS_SERVER protocols for SSLContext.

Remove sha256.tbs-internet.com ssl test

Address ALPN callback changes for OpenSSL 1.1.0f. The latest version behaves like OpenSSL 1.0.2 and no longer aborts handshake.

regrtest: Exclude tzdata from regrtest --all. When running the test suite using --use=all / -u all, exclude tzdata since it makes test_datetime too slow (15-20 min on some buildbots) which then times out on some buildbots. Fix also regrtest command line parser to allow passing -u extralargefile to run test_zipfile64.

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

test_thread: setUp() now uses support.threading_setup() and support.threading_cleanup() to wait until threads complete to avoid random side effects on following tests. Initial patch written by Grzegorz Grzywacz.

Enhanced functions swap_attr() and swap_item() in the test.support module. They now work when delete replaced attribute or item inside the with statement. The old value of the attribute or item (or None if it doesn't exist) now will be assigned to the target of the "as" clause, if there is one.

Use proper command line parsing in _testembed

Disallow -j0 to be combined with -T/-l in regrtest command line arguments.

Fix the tests that bind() a unix socket and raise PermissionError on Android for a non-root user.

Fix the test_socket failures on Android - getservbyname(), getservbyport() and getaddrinfo() are broken on some Android API levels.

Now test.support.rmtree is able to remove unwritable or unreadable directories.

Various caches now are cleared before running every test file.

Fix test_posix for Android where 'id -G' is entirely wrong or missing the effective gid.

regrtest: fix the parser of command line arguments.

Adds _testconsole module to test console input.

Add the support.setswitchinterval() function to fix test_functools hanging on the Android armv7 qemu emulator.

Allow --with-lto to be used on all builds, not just *make profile-opt*.

Remove support for building --without-threads. This option is not really useful anymore in the 21st century. Removing lots of conditional paths allows us to simplify the code base, including in difficult to maintain low-level internal code.

Per PEP 11, support for the IRIX operating system was removed.

Fix compile error when compiling --without-threads. Patch by Masayuki Yamamoto.

Locate msbuild.exe on Windows when building rather than vcvarsall.bat

Support the *disabled* marker in Setup files. Extension modules listed after this marker are not built at all, neither by the Makefile nor by setup.py.

Add `--with-assertions` configure flag to explicitly enable C `assert()` checks. Defaults to off. `--with-pydebug` implies `--with-assertions`.

Fix out-of-tree builds of Python when configured with `--with--dtrace`.

Prevent unnecessary rebuilding of Python during `make test`, `make install` and some other make targets when configured with `--enable-optimizations`.

Don't regenerate generated files based on file modification time anymore: the action is now explicit. Replace `make touch` with `make regen-all`.

Fix `--enable-optimization` didn't work.

sys.version and the platform module python_build(), python_branch(), and python_revision() functions now use git information rather than hg when building from a repo.

Update Windows build and OS X installers to use OpenSSL 1.0.2k.

Prohibit implicit C function declarations: use `-Werror=implicit-function-declaration` when possible (GCC and Clang, but it depends on the compiler version). Patch written by Chi Hsuan Yen.

Remove old Be OS helper scripts.

Set Android compilation and link flags.

Fix implicit declaration of function _setmode. Patch by Masayuki Yamamoto

Removes hard dependency on hg.exe from PCBuild/build.bat

Added missed names to PC/python3.def.

lockf() is available on Android API level 24, but the F_LOCK macro is not defined in android-ndk-r13.

Fix the compilation error that occurs because if_nameindex() is available on Android API level 24, but the if_nameindex structure is not defined.

Do not add the directory for installing C header files and the directory for installing object code libraries to the cross compilation search paths. Original patch by Thomas Petazzoni.

Do not define sys.implementation._multiarch on Android.

Fix out-of-tree building on AIX. Patch by Tristan Carel and Michael Haubenwallner.

Rename --with-optimiations to --enable-optimizations.

Fix missing extensions modules when cross compiling.

Update Windows build and OS X installers to use SQLite 3.14.2.

Update Windows build and OS X installers to use OpenSSL 1.0.2j.

Fix building the _struct module on Cygwin by passing `NULL` instead of `&PyType_Type` to PyVarObject_HEAD_INIT. Patch by Masayuki Yamamoto.

Fix building extensions modules on Cygwin. Patch by Roumen Petrov, based on original patch by Jason Tishler.

Add configure check for siginfo_t.si_band, which Cygwin does not provide. Patch by Masayuki Yamamoto with review and rebase by Erik Bray.

Fixed build with Estonian locale (python-config and distclean targets in Makefile). Patch by Arfrever Frehtes Taifersar Arahesis.

setup.py now detects system libffi with multiarch wrapper.

A full copy of libffi is no longer bundled for use when building _ctypes on non-OSX UNIX platforms. An installed copy of libffi is now required when building _ctypes on such platforms.

Remove redundant include search directory option for building outside the source tree.

Prevent missing 'getentropy' declaration warning on macOS. Patch by Gareth Rees.

Update Windows build to use OpenSSL 1.1.0f

Adds detection of Visual Studio 2017 to distutils on Windows.

zlib is no longer bundled in the CPython source, instead it is downloaded on demand just like bz2, lzma, OpenSSL, Tcl/Tk, and SQLite.

Change to building with MSVC v141 (included with Visual Studio 2017)

os.cpu_count() now returns the correct number of processors on Windows when the number of logical processors is greater than 64.

Pre-build OpenSSL, Tcl and Tk and include the binaries in the build.

Add a missing xmlns to python.manifest so that it matches the schema.

Allow requiring 64-bit interpreters from py.exe using -64 suffix. Contributed by Steve (Gadget) Barnes.

Adds list options (-0, -0p) to py.exe launcher. Contributed by Steve Barnes.

Fix socket deprecation warnings in socketmodule.c. Patch by Segev Finer.

The build process on Windows no longer depends on Subversion, instead pulling external code from GitHub via a Python script. If Python 3.6 is not found on the system (via `py -3.6`), NuGet is used to download a copy of 32-bit Python.

Removes readme.txt from the installer.

winreg does not truncate string correctly (Patch by Eryk Sun)

Deprecate WindowsRegistryFinder and disable it by default

Fixes mishandled buffer reallocation in getpathp.c

Adds signed catalog files for stdlib on Windows.

Enables Unicode for ps1/ps2 and input() prompts. (Patch by Eryk Sun)

Improvements to help manuals on Windows.

launcher.msi has different product codes between 32-bit and 64-bit

Opening CON for write access fails

WindowsConsoleIO readall() fails if first line starts with Ctrl+Z

WindowsConsoleIO fileno() passes wrong flags to _open_osfhandle

_PyIO_get_console_type fails for various paths

Renames Windows path file to ._pth

Windows ._pth file should allow import site

IDLE code context -- fix code update and font update timers. Canceling timers prevents a warning message when test_idle completes.

IDLE - Update non-key options in former extension classes. When applying configdialog changes, call .reload for each feature class. Change ParenMatch so updated options affect existing instances attached to existing editor windows.

IDLE - Improve rstrip entry in doc. Strip trailing whitespace strips more than blank spaces. Multiline string literals are not skipped.

IDLE - make tests pass with zzdummy extension disabled by default.

Document how IDLE runs tkinter programs. IDLE calls tcl/tk update in the background in order to make live interaction and experimentation with tkinter applications much easier.

IDLE -- fix tk entry box tests by deleting first. Adding to an int entry is not the same as deleting and inserting because int('') will fail.

Rearrange IDLE configdialog GenPage into Window, Editor, and Help sections.

IDLE - Add docstrings and tests for outwin subclass of editor. Move some data and functions from the class to module level. Patch by Cheryl Sabella.

IDLE - Do not modify tkinter.message in test_configdialog.

Convert IDLE's built-in 'extensions' to regular features. About 10 IDLE features were implemented as supposedly optional extensions. Their different behavior could be confusing or worse for users and not good for maintenance. Hence the conversion. The main difference for users is that user configurable key bindings for builtin features are now handled uniformly. Now, editing a binding in a keyset only affects its value in the keyset. All bindings are defined together in the system-specific default keysets in config-extensions.def. All custom keysets are saved as a whole in config-extension.cfg. All take effect as soon as one clicks Apply or Ok. The affected events are '<<force-open-completions>>', '<<expand-word>>', '<<force-open-calltip>>', '<<flash-paren>>', '<<format-paragraph>>', '<<run-module>>', '<<check-module>>', and '<<zoom-height>>'. Any (global) customizations made before 3.6.3 will not affect their keyset-specific customization after 3.6.3. and vice versa. Initial patch by Charles Wohlganger.

IDLE: Factor HighPage(Frame) class from ConfigDialog. Patch by Cheryl Sabella.

Add tests for configdialog highlight tab. Patch by Cheryl Sabella.

IDLE: Factor KeysPage(Frame) class from ConfigDialog. The slightly modified tests continue to pass. Patch by Cheryl Sabella.

IDLE -- stop leaks in test_configdialog. Initial patch by Victor Stinner.

Add tests for configdialog keys tab. Patch by Cheryl Sabella.

IDLE: Calltips use *inspect.signature* instead of *inspect.getfullargspec*. This improves calltips for builtins converted to use Argument Clinic. Patch by Louie Lu.

IDLE - Add an outline of a TabPage class in configdialog. Update existing classes to match outline. Initial patch by Cheryl Sabella.

Factor GenPage(Frame) class from ConfigDialog. The slightly modified tests continue to pass. Patch by Cheryl Sabella.

IDLE - Factor FontPage(Frame) class from ConfigDialog. Slightly modified tests continue to pass. Fix General tests. Patch mostly by Cheryl Sabella.

IDLE - Use ttk widgets in ConfigDialog. Patches by Terry Jan Reedy and Cheryl Sabella.

IDLE - Finish rearranging methods of ConfigDialog Grouping methods pertaining to each tab and the buttons will aid writing tests and improving the tabs and will enable splitting the groups into classes.

IDLE -- Factor a VarTrace class out of ConfigDialog. Instance tracers manages pairs consisting of a tk variable and a callback function. When tracing is turned on, setting the variable calls the function. Test coverage for the new class is 100%.

IDLE: Add more tests for General tab.

IDLE - Improve configdialog font page and tests. In configdialog: Document causal pathways in create_font_tab docstring. Simplify some attribute names. Move set_samples calls to var_changed_font (idea from Cheryl Sabella). Move related functions to positions after the create widgets function. In test_configdialog: Fix test_font_set so not order dependent. Fix renamed test_indent_scale so it tests the widget. Adjust tests for movement of set_samples call. Add tests for load functions. Put all font tests in one class and tab indent tests in another. Except for two lines, these tests completely cover the related functions.

IDLE -- Add more configdialog font page tests.

IDLE: replace 'colour' with 'color' in configdialog.

Add tests for idlelib.config.IdleConf. Increase coverage from 46% to 96%. Patch by Louie Lu.

Document coverage details for idlelib tests. Add section to idlelib/idle-test/README.txt. Include check that branches are taken both ways. Exclude IDLE-specific code that does not run during unit tests.

IDLE: Document ConfigDialog tk Vars, methods, and widgets in docstrings This will facilitate improving the dialog and splitting up the class. Original patch by Cheryl Sabella.

IDLE: Add tests for ConfigParser subclasses in config. Patch by Louie Lu.

IDLE: Add docstrings to browser.py. Patch by Cheryl Sabella.

IDLE: Remove unused variables in configdialog. One is a duplicate, one is set but cannot be altered by users. Patch by Cheryl Sabella.

IDLE: In Settings dialog, select font with Up, Down keys as well as mouse. Initial patch by Louie Lu.

IDLE: call config.IdleConf.GetUserCfgDir only once.

IDLE: Factor ConfigChanges class from configdialog, put in config; test. * In config, put dump test code in a function; run it and unittest in 'if __name__ == '__main__'. * Add class config.ConfigChanges based on changes_class_v4.py on bpo issue. * Add class test_config.ChangesTest, partly using configdialog_tests_v1.py. * Revise configdialog to use ConfigChanges; see tracker msg297804. * Revise test_configdialog to match configdialog changes. * Remove configdialog functions unused or moved to ConfigChanges. Cheryl Sabella contributed parts of the patch.

IDLE: configdialog - Add docstrings and fix comments. Patch by Cheryl Sabella.

IDLE: Improve textview with docstrings, PEP8 names, and more tests. Patch by Cheryl Sabella.

IDLE: Make several improvements to parenmatch. Add 'parens' style to highlight both opener and closer. Make 'default' style, which is not default, a synonym for 'opener'. Make time-delay work the same with all styles. Add help for config dialog extensions tab, including help for parenmatch. Add new tests. Original patch by Charles Wohlganger.

IDLE: add docstrings to grep module. Patch by Cheryl Sabella

IDLE's basic custom key entry dialog now detects duplicates properly. Original patch by Saimadhav Heblikar.

IDLE no longer deletes a character after commenting out a region by a key shortcut. Add `return 'break'` for this and other potential conflicts between IDLE and default key bindings.

Review and change idlelib.configdialog names. Lowercase method and attribute names. Replace 'colour' with 'color', expand overly cryptic names, delete unneeded underscores. Replace `import *` with specific imports. Patches by Cheryl Sabella.

IDLE: Verify user-entered key sequences by trying to bind them with tk. Add tests for all 3 validation functions. Original patch by G Polo. Tests added by Cheryl Sabella.

Fix several problems with IDLE's autocompletion box. The following should now work: clicking on selection box items; using the scrollbar; selecting an item by hitting Return. Hangs on MacOSX should no longer happen. Patch by Louie Lu.

Add doc subsubsection about IDLE failure to start. Popup no-connection message directs users to this section.

Fix reference leaks in IDLE tests. Patches by Louie Lu and Terry Jan Reedy.

Add docstrings for textview.py and use PEP8 names. Patches by Cheryl Sabella and Terry Jan Reedy.

Help-about: use pep8 names and add tests. Increase coverage to 100%. Patches by Louie Lu, Cheryl Sabella, and Terry Jan Reedy.

Add _utest option to textview; add new tests. Increase coverage to 100%. Patches by Louie Lu and Terry Jan Reedy.

IDLE colors f-string prefixes (but not invalid ur prefixes).

Add 10% to coverage of IDLE's test_configdialog. Update and augment description of the configuration system.

gdb integration commands (py-bt, etc.) work on optimized shared builds now, too. PEP 523 introduced _PyEval_EvalFrameDefault which inlines PyEval_EvalFrameEx on non-debug shared builds. This broke the ability to use py-bt, py-up, and a few other Python-specific gdb integrations. The problem is fixed by only looking for _PyEval_EvalFrameDefault frames in python-gdb.py. Original patch by Bruno "Polaco" Penteado.

Added the slice index converter in Argument Clinic.

Argument Clinic now uses the converter *bool(accept={int})* rather than *int* for semantical booleans. This avoids repeating the default value for Python and C and will help in converting to *bool* in future.

python-gdb.py now supports also `method-wrapper` (`wrapperobject`) objects.

Fix python-gdb.py didn't support new dict implementation.

The pybench and pystone microbenchmark have been removed from Tools. Please use the new Python benchmark suite https://github.com/python/performance which is more reliable and includes a portable version of pybench working on Python 2 and Python 3.

The zipfile module CLI now prints usage to stderr. Patch by Stephen J. Turnbull.

Added the `Py_UNREACHABLE()` macro for code paths which are never expected to be reached. This and a few other useful macros are now documented in the C API manual.

Remove own implementation for thread-local storage. CPython has provided the own implementation for thread-local storage (TLS) on Python/thread.c, it's used in the case which a platform has not supplied native TLS. However, currently all supported platforms (Windows and pthreads) have provided native TLS and defined the Py_HAVE_NATIVE_TLS macro with unconditional in any case.

PyUnicode_AsWideCharString() now raises a ValueError if the second argument is NULL and the wchar_t* string contains null characters.

Deprecate PyOS_AfterFork() and add PyOS_BeforeFork(), PyOS_AfterFork_Parent() and PyOS_AfterFork_Child().

The type of results of PyThread_start_new_thread() and PyThread_get_thread_ident(), and the id parameter of PyThreadState_SetAsyncExc() changed from "long" to "unsigned long".

Function PySlice_GetIndicesEx() is deprecated and replaced with a macro if Py_LIMITED_API is not set or set to the value between 0x03050400 and 0x03060000 (not including) or 0x03060100 or higher. Added functions PySlice_Unpack() and PySlice_AdjustIndices().

Fixed the declaration of some public API functions. PyArg_VaParse() and PyArg_VaParseTupleAndKeywords() were not available in limited API. PyArg_ValidateKeywordArguments(), PyArg_UnpackTuple() and Py_BuildValue() were not available in limited API of version < 3.3 when PY_SSIZE_T_CLEAN is defined.

The result of PyUnicode_AsUTF8AndSize() and PyUnicode_AsUTF8() is now of type `const char *` rather of `char *`.

All stable API extensions added after Python 3.2 are now available only when Py_LIMITED_API is set to the PY_VERSION_HEX value of the minimum Python version supporting this API.

The index parameters *start* and *end* of PyUnicode_FindChar() are now adjusted to behave like `str[start:end]`.

PyUnicode_CompareWithASCIIString() now never raises exceptions.

The fields name and doc of structures PyMemberDef, PyGetSetDef, PyStructSequence_Field, PyStructSequence_Desc, and wrapperbase are now of type `const char *` rather of `char *`.

Private variable _Py_PackageContext is now of type `const char *` rather of `char *`.

Compiler warnings are now emitted if use most of deprecated functions.

Deprecated undocumented functions PyUnicode_AsEncodedObject(), PyUnicode_AsDecodedObject(), PyUnicode_AsDecodedUnicode() and PyUnicode_AsEncodedUnicode().