

The default warning filter list now starts with a "default::DeprecationWarning: __main__" entry, so deprecation warnings are once again shown by default in single-file scripts and at the interactive prompt.

`__class_getitem__` is now an automatic class method.

Add AIX uuid library support for RFC4122 using `uuid_create()` in `libc.a`

Fix the compilation failure on AIX after the `f_fsid` field has been added to the object returned by `os.statvfs()` (issue #32143). Original patch by Michael Felt.

Make MRO computation faster when a class inherits from a single base.

The error message of a `TypeError` raised when unpack non-iterable is now more specific.

The `__debug__` constant is now optimized out at compile time. This fixes also bpo-22091.

The `:option: -R` option now turns on hash randomization when the `:envvar: PYTHONHASHSEED` environment variable is set to 0. Previously, the option was ignored. Moreover, `sys.flags.hash_randomization` is now properly set to 0 when hash randomization is turned off by `PYTHONHASHSEED=0`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 77); [backlink](#)

Unknown interpreted text role "option".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 77); [backlink](#)

Unknown interpreted text role "envvar".

The optimizer is now protected from spending much time doing complex calculations and consuming much memory for creating large constants in constant folding. Increased limits for constants that can be produced in constant folding.

Fix an unnecessary `ifdef` in the include of `VersionHelpers.h` in `socketmodule` on Windows.

Implement `TracebackType.__new__` to allow Python-level creation of traceback objects, and make `TracebackType.tb_next` mutable.

Don't byte swap the input keys to the SipHash algorithm on big-endian platforms. This should ensure siphash gives consistent results across platforms.

Improve the error message logic for object `__new__` and object `__init__`. Patch by Sanyam Khurana.

`-b` and `-bb` now inject `'default::BytesWarning'` and `error::BytesWarning` entries into `sys.warnoptions`, ensuring that they take precedence over any other warning filters configured via the `-W` option or the `PYTHONWARNINGS` environment variable.

`-X dev` now injects a `'default'` entry into `sys.warnoptions`, ensuring that it behaves identically to actually passing `-Wdefault` at the command line.

Add a new UTF-8 mode: implementation of the [PEP 540](#).

[PEP 560](#): Add support for `__mro_entries__` and `__class_getitem__`. Implemented by Ivan Levkivskyi.

[PEP 562](#): Add support for module `__getattr__` and `__dir__`. Implemented by Ivan Levkivskyi.

The `atexit` module now has its callback stored per interpreter.

Implement [PEP 552](#) (Deterministic pycs). Python now supports invalidating bytecode cache files based on a source content hash rather than source last-modified time.

Move constant folding from bytecode layer to AST layer. Original patch by Eugene Toder.

Now that dict is defined as keeping insertion order, drop `OrderedDict` and just use plain dict.

Add params to `dataclasses.make_dataclasses()`: `init`, `repr`, `eq`, `order`, `hash`, and `frozen`. Pass them through to `dataclass()`.

Make type information optional on `dataclasses.make_dataclass()`. If omitted, the string `'typing.Any'` is used.

Add `dataclasses.is_dataclass(obj)`, which returns `True` if `obj` is a dataclass or an instance of one.

Improve `frame.repr()` to mention filename, code name and current line number.

asyncio: Implement `loop.start_tls()`

Return the new file descriptor (i.e., the second argument) from `os.dup2`. Previously, `None` was always returned.

`functools.lru_cache` uses less memory (3 words for each cached key) and takes about 1/3 time for cyclic GC.

Prevent Python crash from happening when `Future._log_traceback` is set to `True` manually. Now it can only be set to `False`, or a `ValueError` is raised.

asyncio: Add `Task.get_loop()` and `Future.get_loop()`

Don't unsubscribe signals in asyncio UNIX event loop on interpreter shutdown.

Make `asyncio.Task.set_exception()` and `set_result()` raise `NotImplementedError`. `Task._step()` and `Future.__await__()` raise proper exceptions when they are in an invalid state, instead of raising an `AssertionError`.

Optimize `asyncio.iscoroutine()` and `loop.create_task()` for non-native coroutines (e.g. `async/await` compiled with Cython).

'`loop.create_task(python_coroutine)`' used to be 20% faster than '`loop.create_task(cython_coroutine)`'. Now, the latter is as fast.

`asyncio.transport.resume_reading()` and `pause_reading()` are now idempotent. New `transport.is_reading()` method is added.

Optimize `asyncio.gather()`; now up to 15% faster.

Use fastpath in `asyncio.sleep` if `delay<0` (2x boost)

Optimize `asyncio.Future.schedule/add/remove callback`. The optimization shows 3-6% performance improvements of `async/await` code.

Fix `socket.settimeout()` and `socket.setblocking()` to keep `socket.type` as is. Fix `socket.socket()` constructor to reset any bit flags applied to socket's type. This change only affects OSes that have `SOCK_NONBLOCK` and/or `SOCK_CLOEXEC`.

Add `:class:'importlib.abc.ResourceReader'` as an ABC for loaders to provide a unified API for reading resources contained within packages. Also add `:mod:'importlib.resources'` as the port of `importlib_resources`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 406); [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) 3.7.0a4.rst, line 406); [backlink](#)

Unknown interpreted text role "mod".

Implement `asyncio.create_task(coro)` shortcut

Convert `asyncio` functions that were documented as coroutines to coroutines. Affected functions: `loop.sock_sendall`, `loop.sock_recv`, `loop.sock_accept`, `loop.getaddrinfo`, `loop.getnameinfo`.

`:func:'urlib.parse.urlsplit()'` does not convert zone-id (scope) to lower case for scoped IPv6 addresses in hostnames now.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 437); [backlink](#)

Unknown interpreted text role "func".

Fix `bdist_wininst` of distutils for CRT v142: it binary compatible with CRT v140.

Fix `stop_serving` in `asyncio` proactor loop kill all listening servers

`:func:'re.sub()'` now replaces empty matches adjacent to a previous non-empty match.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 466); [backlink](#)

Unknown interpreted text role "func".

Abort `asyncio.SSLProtocol` connection if handshake not complete within 10 seconds.

Implement `asyncio.run()`.

Revert incorrect fix based on misunderstanding of `_Py_PyAtExit()` semantics.

Implement `asyncio._get_running_loop()` and `get_event_loop()` in C. This makes them 4x faster.

Implement `asyncio.current_task()` and `asyncio.all_tasks()`. Add helpers intended to be used by alternative task implementations: `asyncio._register_task`, `asyncio._enter_task`, `asyncio._leave_task` and `asyncio._unregister_task`. Deprecate `asyncio.Task.current_task()` and `asyncio.Task.all_tasks()`.

A single empty field is now always quoted when written into a CSV file. This allows to distinguish an empty row from a row consisting of a single empty field. Patch by Licht Takeuchi.

Raise `NotImplementedError` instead of `SystemError` on platforms where `chmod(..., follow_symlinks=False)` is not supported. Patch by Anthony Sottile.

New argument `warn_on_full_buffer` to `signal.set_wakeup_fd` lets you control whether Python prints a warning on stderr when the wakeup fd buffer overflows.

The `fpectl` library has been removed. It was never enabled by default, never worked correctly on x86-64, and it changed the Python ABI in ways that caused unexpected breakage of C extensions.

Move `asyncio.test_utils` to `test.test_asyncio`.

Remove `asyncio.async()` function.

Add `asyncio.get_running_loop()` function.

All class and static methods of builtin types now are correctly classified by `inspect.classify_class_attrs()` and grouped in `pydoc` output. Added `types.ClassMethodDescriptorType` for unbound class methods of builtin types.

Deprecate `yield from lock`, `await lock`, `with (yield from lock)` and `with await lock` for `asyncio` synchronization primitives.

Changed MIME type of `.bmp` from `'image/x-ms-bmp'` to `'image/bmp'`

Convert `asyncio` to use `async/await` syntax. Old styled `yield from` is still supported too.

Add support to run modules with `pdb`

`functools.singledispatch` now supports registering implementations using type annotations.

Added new alternate constructors `meth:`datetime.datetime.fromisoformat``, `meth:`datetime.time.fromisoformat`` and `meth:`datetime.date.fromisoformat`` as the inverse operation of each classes's respective `isoformat` methods.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 656); [backlink](#)

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 656); [backlink](#)

Unknown interpreted text role "meth".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 656); [backlink](#)

Unknown interpreted text role "meth".

The `getnode()` ip getter now uses `'ip link'` instead of `'ip link list'`.

`os.statvfs()` includes the `f_fsid` field from `statvfs(2)`

Fix `ctypes.util.find_library()` for AIX by implementing `ctypes._aix.find_library()` Patch by: Michael Felt

The pickler now uses less memory when serializing large bytes and str objects into a file. Pickles created with protocol 4 will require less memory for unpickling large bytes and str objects.

Ensure `TCP_NODELAY` is set on Linux. Tests by Victor Stinner.

`ast.literal_eval()` is now more strict. Addition and subtraction of arbitrary numbers no longer allowed.

Importing native path module (`posixpath`, `ntpath`) now works even if the `os` module still is not imported.

Add `contextlib.AbstractAsyncContextManager`. Patch by Jelle Zijlstra.

Fix deadlocks in `:class:`concurrent.futures.ProcessPoolExecutor`` when task arguments or results cause pickling or unpickling errors. This should make sure that calls to the `:class:`ProcessPoolExecutor`` API always eventually return.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 744); [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) 3.7.0a4.rst, line 744); [backlink](#)

Unknown interpreted text role "class".

`TextIOWrapper.reconfigure()` supports changing *encoding*, *errors*, and *newline*.

Add `get_loop()` method to `Server` and `AbstractServer` classes.

Fix `faulthandler_suppress_crash_report()` used to prevent core dump files when testing crashes. `getrlimit()` returns zero on success.

Adjust C locale coercion testing for the empty locale and POSIX locale cases to more readily adjust to platform dependent behaviour.

Implement support for *subprocess.Popen(close_fds=True)* on Windows. Patch by Segev Finer.

2to3 and lib2to3 can now read pickled grammar files using `pkgutil.get_data()` rather than probing the filesystem. This lets 2to3 and lib2to3 work when run from a zipfile.

`Py_Initialize()` doesn't reset the memory allocators to default if the `PYTHONMALLOC` environment variable is not set.

Undocumented C API for `OrderedDict` has been excluded from the limited C API. It was added by mistake and actually never worked in the limited C API.

Moved the `pygetopt.h` header into `internal/`, since it has no public APIs.

`:c:func:'Py_SetProgramName'` and `:c:func:'Py_SetPythonHome'` now take the `const wchar *` arguments instead of `wchar *`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 845); [backlink](#)

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Misc\NEWS.d\ (cpython-main) (Misc) (NEWS.d) 3.7.0a4.rst, line 845); [backlink](#)

Unknown interpreted text role "c:func".