

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 deprecated. Patch by Oren Milman.

The constructor of `range_iterator` now checks that `step` is not 0. 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.

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

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

Upgrade internal unicode databases to Unicode version 9.0.0.

Fix a regression in `zipimport.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.

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

The disassembler now decodes `FORMAT_VALUE` argument.

Fixed writing ZIP files that starts not from the start of the file. Offsets in ZIP file now are relative to the start of the archive in conforming to the specification.

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

remove `statistics.geometric_mean` and defer until 3.7.

`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.

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.

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 Ššš¼ŃĖĜµĜ¹Ĝ±ĜµŃĖĜ³ ĞœĜ°ŃĖĜ°.

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 Ššš¼ŃĖĜµĜ¹Ĝ½Ĝ±ĜµŃĖĜ³ ĞœĜ°ŃĖĜ°.

Prohibit Tasks to await on themselves.

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

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

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.



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

Adds `_testconsole` module to test console input.

