Backward Incompatible Changes

- PHP Core
  - "array" and "callable" alias name
  - Loosely comparing uncomparable objects
  - Return value of gc_collect_cycles
  - Substitute static keyword in final subclass
  - Tick handlers
  - Traits binding
  - Errors during compilation and class linking
  - Exceptions thrown by user-defined error handler
  - Attribute apply error during compilation
  - disable_classes INI setting
  - Destructuring non-array values
  - Warnings related to cast
- Bzip2
- DOM
- FileInfo
- Intl
- LDAP
- MBString
- MySQLi
- ODBC
- Opcache
- PCNTL
- PCRE
- PDO
- PDO_FIREBIRD

- PDO_SQLITE
- POSIX
- Reflection
- Session
- SimpleXML
- SNMP
- SOAP
- Sockets
- SPL
- Standard
- Tidy

PHP Core

`"array"` and `"callable"` alias name

It is no longer possible to use `"array"` and `"callable"` as class alias names in `class_alias`

Loosely comparing uncomparable objects

Loosely comparing uncomparable objects (e.g. enums, `CurlHandle` and other internal classes) to booleans was previously inconsistent. If compared to a boolean literal `$object == true`, it would behave the same way as `(bool)$object`. If compared to a statically unknown value `$object == $true`, it would always return ENTITY. This behavior was consolidated to always follow the behavior of `(bool)$object`.

Return value of gc_collect_cycles

The return value of `gc_collect_cycles` no longer includes strings and resources that were indirectly collected through cycles.

Substitute static keyword in final subclass

It is now allowed to substitute `static` with `self` or the concrete class name in final subclasses.

Tick handlers

The tick handlers are now deactivated after all shutdown functions, destructors have run and the output handlers have been cleaned up.

Traits binding

Traits are now bound before the parent class. This is a subtle behavioral change, but should more closely match user expectations.

Errors during compilation and class linking

Errors emitted during compilation and class linking are now always delayed and handled after compilation or class linking. Fatal errors emitted during compilation or class linking cause any delayed errors to be handled immediately, without calling user-defined error handlers.

Exceptions thrown by user-defined error handler

Exceptions thrown by user-defined error handlers when handling class linking errors are not promoted to fatal errors anymore and do not prevent linking.

Attribute apply error during compilation

Applying `#[\Attribute]` to an abstract class, enum, interface, or trait triggers an error during compilation. Previously, the attribute could be added, but when ReflectionAttribute::newInstance was called an error would be thrown. The error can be delayed from compilation to runtime using the new `#[\DelayedTargetValidation]` attribute.

disable_classes INI setting

The disable_classes INI setting has been removed as it causes various engine assumptions to be broken.

Destructuring non-array values

Destructuring non-array values (other than ENTITY) using `[]` or `list` now emits a warning.

Warnings related to cast

A warning is now emitted when casting floats (or strings that look like floats) to int if they cannot be represented as one. This affects explicit int casts and implicit int casts.

A warning is now emitted when casting NAN to other types.

Bzip2

`bzcompress` now throws a `ValueError` when `block_size` is not between 1 and 9.

`bzcompress` now throws a `ValueError` when `work_factor` is not between 0 and 250.

DOM

Cloning a `DOMNamedNodeMap`, `DOMNodeList`, `Dom\NamedNodeMap`, `Dom\NodeList`, `Dom\HTMLCollection`, and `Dom\DtdNamedNodeMap` now fails. This never actually resulted in a working object, therefore no impact is expected.

FileInfo

`finfo_file` and finfo::file now throws a ValueError instead of a TypeError when `filename` contains nul bytes. This aligns the type of Error thrown to be consistent with the rest of the language.

Intl

The extension now requires at least ICU 57.1.

IntlDateFormatter::setTimeZone/`datefmt_set_timezone` now throws an IntlException on uninitialised classes/clone failures.

All `Locale` methods now throw a ValueError when the locale argument contain null bytes.

The behaviour of `Collator::SORT_REGULAR` with respect to handling numeric strings is now aligned with the behaviour of `SORT_REGULAR` in ext/standard.

LDAP

`ldap_get_option` and `ldap_set_option` now throw a ValueError when passing an invalid option.

MBString

Unicode data tables have been updated to Unicode 17.0

MySQLi

Calling the mysqli constructor on an already-constructed object is now no longer possible and throws an Error.

ODBC

ODBC now assumes that at least ODBC 3.5 functionality is available. The ODBCVER definition and build system flags to control it have been removed.

ODBC no longer has build flags to build against specific drivers (except for DB2) and removes special cases for those drivers. It is strongly recommended to use a driver manager like iODBC or unixODBC on non-Windows.

Opcache

The Opcache extension is now always built into the PHP binary and is always loaded. The INI directives opcache.enable and opcache.enable_cli are still honored.

The `--enable-opcache`/`--disable-opcache` configure flags have been removed, and the build does not produce `opcache.so` or `php_opcache.dll` objects anymore.

Using `zend_extension=opcache.so` or `zend_extension=php_opcache.dll` INI directives will emit a warning.

PCNTL

`pcntl_exec` now throws ValueError when entries of the `args` parameter contain null bytes.

`pcntl_exec` now throws ValueError when entries or keys of the `env_vars` parameter contain null bytes.

PCRE

The extension is compiled without semi-deprecated PCRE2_EXTRA_ALLOW_LOOKAROUND_BSK compile option.

PDO

The constructor arguments set in conjunction with `PDO::FETCH_CLASS` now follow the usual CUFA (`call_user_func_array`) semantics. This means string keys will act like a named argument. Moreover, automatic wrapping for by-value arguments passed to a by-ref parameter has been removed, and the usual `E_WARNING` about this is now emitted. To pass a variable by-ref to a constructor argument use the general array value reference assignment: `$ctor_args = [ENTITY$valByRef]`

Attempting to call PDOStatement::setFetchMode during a call to PDOStatement::fetch, PDOStatement::fetchObject, PDOStatement::fetchAll, for example using tricks such as passing the statement object as a constructor argument when fetching into an object, will now throw an Error.

The value of the constants `PDO::FETCH_GROUP`, `PDO::FETCH_UNIQUE`, `PDO::FETCH_CLASSTYPE`, `PDO::FETCH_PROPS_LATE`, and `PDO::FETCH_SERIALIZE` have changed.

A ValueError is now thrown if `PDO::FETCH_PROPS_LATE` is used with a fetch mode different than `PDO::FETCH_CLASS`, consistent with other fetch flags.

A ValueError is now thrown if `PDO::FETCH_INTO` is used as a fetch mode in PDOStatement::fetchAll, similar to `PDO::FETCH_LAZY`.

PDO_FIREBIRD

A ValueError is now thrown when trying to set a cursor name that is too long on a `PDOStatement` resulting from the Firebird driver.

PDO_SQLITE

SQLite PDO::quote will now throw an exception or emit a warning, depending on the error mode, if the string contains a null byte.

PDO::sqliteCreateCollation will now throw an exception if the callback has the wrong return type, making it more in line with Pdo\Sqlite::createCollation behavior.

POSIX

`posix_kill` now throws a ValueError when the process_id argument is lower or greater than what the platform supports (signed integer or long range), `posix_setpgid` now throws a ValueError when the process_id or the process_group_id is lower than zero or greater than what the platform supports.

`posix_setrlimit` now throws a ValueError when the hard_limit or soft_limit arguments are lower than -1 or if soft_limit is greater than hard_limit.

Reflection

ReflectionAttribute::newInstance can now throw errors for internal attributes if the attribute was applied on an invalid target and the error was delayed from compile time to runtime via the #[\DelayedTargetValidation] attribute.

Session

Attempting to write session data where `$_SESSION` has a key containing the pipe character (`|`) will now emit a warning instead of silently failing.

`session_start` is stricter in regard to the options argument. It now throws a ValueError if the array is not a hashmap, or a TypeError if the read_and_close value is not a valid type compatible with int.

SimpleXML

Passing an XPath expression that returns something other than a node set to SimpleXMLElement::xpath will now emit a warning and return ENTITY, instead of silently failing and returning an empty array.

SNMP

`snmpget`, `snmpset`, `snmp2_get`, `snmp2_set`, `snmp3_get`, `snmp3_set` and SNMP::__construct now throw a ValueError when the hostname is too large, contains any null byte or if the port is given when negative or greater than 65535, timeout and retries values are lower than -1 or too large.

SOAP

SoapClient::__doRequest now accepts a new, optional `uriParserClass` parameter accepting string or ENTITY arguments. ENTITY represents the original (`parse_url`) based method, while the new backends will be used when passing either `Uri\Rfc3986\Uri` or `Uri\WhatWg\Url`.

Sockets

`socket_create_listen`, `socket_bind` and `socket_sendto` now throw a ValueError if the port is lower than 0 or greater than 65535, and also if any of the hints array entries are indexed numerically.

`socket_addrinfo_lookup` now throws a TypeError if any of the hints values cannot be cast to int and can throw a ValueError if any of these values overflow.

`socket_set_option` with `MCAST_LEAVE_GROUP`/`MCAST_LEAVE_SOURCE_GROUP` options now throw an exception if the value isn't a valid object or array.

`socket_set_option` with multicast context now throws a ValueError when the created socket is not of `AF_INET`/`AF_INET6` family.

SPL

`ArrayObject` no longer accepts enums, as modifying the $name or $value properties can break engine assumptions.

SplFileObject::fwrite's parameter `length` is now nullable. The default value changed from `0` to ENTITY.

Standard

Using a printf-family function with a formatter that did not specify the precision previously incorrectly reset the precision instead of treating it as a precision of 0.

Passing an integer `0` as the `locales` argument to `setlocale` is no longer supported and now throws a TypeError.

Tidy

tidy::__construct, tidy::parseFile, tidy::parseString now throws a ValueError if the configuration contains an invalid value or attempts to set a read-only internal entry, and a TypeError if a configuration key is not a string.

□□□ #2
□ □□□ □□□ 2026-10-31 17:36:45 UTC
□ □□□ □□□ 2026-10-31 17:36:46 UTC